ABSTRACT

ASTRA: A FINITE ELEMENT STRUCTURAL ANALYSIS PROGRAM FOR THE APPLE II MICROCOMPUTER

by

Ricardo Crespo

December 1981

This thesis describes a new finite element structural analysis computer program "ASTRA" developed by the author for use on the APPLE II microcomputer. The program is written in Applesoft Basic and is based on the displacement method of matrix structural analysis. It has a capacity of twenty five nodes and elements, and two types of structural elements: a space rod and a space beam. Along with a review of the basic applicable structural theorems needed, a thorough discussion of the program is presented. Description of special purpose subroutines, methods of data handling/ storage, and suggestions on possible improvement are also included. It is hoped that this program can serve as a base from which more advanced programs can be developed for use with the APPLE II microcomputer.

ASTRA: A FINITE ELEMENT STRUCTURAL ANALYSIS PROGRAM FOR THE APPLE II MICROCOMPUTER

A THESIS

Presented to the Department of Mechanical Engineering

California State University, Long Beach

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

By Ricardo Crespo

December 1981

WE, THE UNDERSIGNED MEMBERS OF THE COMMITTEE,

HAVE APPROVED THIS THESIS

ASTRA: A FINITE ELEMENT STRUCTURAL ANALYSIS PROGRAM FOR THE APPLE II MICROCOMPUTER

By

Ricardo Crespo

COMMITTEE MEMBERS

Bruce J. Torby

Bruce J. Torby, Ph.D. (Chair) Mechanical Engineering

Hillar Unt

Hillar Unt, Ph.D. Mechanical Engineering

Ching H. Tsao

Ching H. Tsao, Ph.D. Mechanical Engineering

ACCEPTED AND APPROVED ON BEHALF OF THE UNIVERSITY

Richard C. Potter

Richard C. Potter, Ph.D.
Dean, School of Engineering

California State University, Long Beach

December 1981

ACKNOWLEDGEMENTS

During the preparation of this paper, a number of persons aided me extensively. I would like to extend my appreciation to them and especially Bruce J. Torby, Ph.D. for his guidance and tutelage, the management of The Aerospace Corporation for their understanding and support, Urania Yuan, Karen Carter, Faye Ernest, and Regina Wallace who typed it, and last but not least, my Parents, Godmother and Patty Bland for their moral support.

CONTENTS

TABLES

FIGURES

Chapter 1

INTRODUCTION

The rapid advances of science in the field of data processing, particularly the introduction of microcomputers, has made available low cost computer equipment to engineers and scientists. These computers can greatly enhance and speed the solution of complex engineering problems.

In the field of structural analysis, the idea of utilizing computers to obtain the solution of complex structural problems is not new, but the majority of modern finite elements structural analysis programs are available only on large and expensive computer systems. It is felt that modern microcomputers are powerful enough to warrant the creation of software for the analysis of structures using the finite element method. This study is intended to validate this concept by developing a finite element structural analysis program for use on the APPLE II Plus microcomputer. The program named ASTRA, an abbreviation for "Analyzing Structures with APPLE," is not intended to compete with the more advanced structural programs available on main-frame computers. Rather, it has been designed to provide an

alternative program for use by small companies which do not have the capital to employ one of the larger programs, by field engineers to use in remote locations where access to main-frame computers is limited, or by educational institutions teaching the finite element method.

ASTRA is made up the pre-processor IASTRA (Input to ASTRA) and the main program ASTRA. IASTRA was written to obtain all the information required for the analysis of structures and to generate the input data for use in ASTRA. In this manner, the amount of time needed to train a future user of the program should be minimized and errors reduced. Extra care was also devoted to make the program ASTRA user oriented, i.e., the input of information necessary for the running of the program was made as easy as possible in order to make the program convenient to use with little formal training.

Chapter 2

BASIC STRUCTURAL THEOREMS

Introduction

The methods of analysis discussed here are limited in application to elastic structures. The following assumptions are made for each element:

1. The body is elastic
2. The material properties of the body are homogeneous and isotropic
3. The deformations are assumed to be infinitesimal

The Theory of Elasticity forms the basis for performing analysis of structures. In order to define the various terms involved in the displacement and force methods of analysis, the fundamentals of the Theory of Elasticity and basic structural theorems will first be quickly reviewed.

Three conditions must always be met in any analysis of the internal forces and deformation in a structure, these are:

1. Equilibrium of forces must exist
2. Laws of material behavior must be obeyed, and
3. The displacement results must be compatible

The first condition requires that the internal forces balance the applied external loads. This condition alone is sufficient to enable the solution of statically determinate problems, however, it yields insufficient information to enable the analysis of redundant structures. Under those circumstances the laws of material behavior which in problems of linear elasticity, neglecting temperature changes, reduce to Hooke's Law, and the conditions of compatibility must be invoked in order to complete the analysis.

EQUATIONS OF EQUILIBRIUM

If we consider the cube of Figure 1 and express the equilibrium of forces acting on it, in the x direction one has:

$$\begin{aligned} X\,dx\,dy\,dz \quad & -\sigma_x dy\,dz + \left(\sigma_x + \frac{\partial \sigma_x}{\partial x} dx\right) dy\,dz - \tau_{yx} dz\,dx \\ & + \left(\tau_{yx} + \frac{\partial \tau_{yx}}{\partial y} dy\right) dx\,dz - \tau_{zx} dy\,dx + \left(\tau_{zx} + \frac{\partial \tau_{zx}}{\partial z} dz\right) dy\,dx = 0 \end{aligned} \tag{2.1}$$

where X is the force in the x direction per unit volume. Cancelling dx dy dz we obtain:

$$\frac{\partial \sigma_x}{\partial x} + \frac{\partial \tau_{xy}}{\partial y} + \frac{\partial \tau_{xz}}{\partial z} + X = 0 \tag{2.2}$$

Similar equations can be obtained for the equilibrium of forces in the y and z directions.

$$\frac{\partial \tau_{yx}}{\partial x} + \frac{\partial \sigma_y}{\partial y} + \frac{\partial \tau_{yz}}{\partial z} + Y = 0 \tag{2.3}$$

$$\frac{\partial \tau_{zx}}{\partial x} + \frac{\partial \tau_{zy}}{\partial y} + \frac{\partial \sigma_z}{\partial z} + Z = 0 \tag{2.4}$$

Equations (2.2) through (2.4) are Navier's equations of equilibrium for an elastic solid.

Stress-Strain Relationships

The relationship between stress and strain, or the law of material behavior for a linear elastic body below its yield point are given as:

$$\begin{aligned} \epsilon_x &= \frac{1}{E}[\sigma_x - \nu(\sigma_y + \sigma_z)] \\ \epsilon_y &= \frac{1}{E}[\sigma_y - \nu(\sigma_z + \sigma_x)] \\ \epsilon_z &= \frac{1}{E}[\sigma_z - \nu(\sigma_x + \sigma_y)] \end{aligned} \tag{2.5}$$

and

$$\begin{aligned} \gamma_{xy} &= \frac{\tau_{xy}}{G} \\ \gamma_{yz} &= \frac{\tau_{yz}}{G} \\ \gamma_{zx} &= \frac{\tau_{zx}}{G} \end{aligned} \tag{2.6}$$

Equations of Compatibility

The components of strain in terms of the components of elastic displacement can be presented as follows:

$$\epsilon_x = \frac{\partial u}{\partial x}$$
$$\epsilon_y = \frac{\partial v}{\partial y} \qquad (2.7)$$
$$\epsilon_z = \frac{\partial w}{\partial z}$$

and

$$\gamma_{xy} = \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}$$
$$\gamma_{yz} = \frac{\partial v}{\partial z} + \frac{\partial w}{\partial y} \qquad (2.8)$$
$$\gamma_{zx} = \frac{\partial w}{\partial x} + \frac{\partial u}{\partial z}$$

The components of strain are not independent but are related by relationships called compatibility equations. From Equations (2.7) and (2.8) it follows that

$$
\begin{aligned}
\frac{\partial^2 \epsilon_x}{\partial y^2} + \frac{\partial^2 \epsilon_y}{\partial x^2} &= \frac{\partial^2 \gamma_{xy}}{\partial x \partial y} \\
\frac{\partial^2 \epsilon_y}{\partial z^2} + \frac{\partial^2 \epsilon_z}{\partial y^2} &= \frac{\partial^2 \gamma_{yz}}{\partial y \partial z} \qquad (2.9) \\
\frac{\partial^2 \epsilon_z}{\partial x^2} + \frac{\partial^2 \epsilon_x}{\partial z^2} &= \frac{\partial^2 \gamma_{zx}}{\partial z \partial x}
\end{aligned}
$$

and

$$
\begin{aligned}
\frac{\partial}{\partial x}\left[\frac{\partial \gamma_{xy}}{\partial z} - \frac{\partial \gamma_{yz}}{\partial x} + \frac{\partial \gamma_{zx}}{\partial y}\right] &= 2\frac{\partial^2 \epsilon_x}{\partial y \partial z} \\
\frac{\partial}{\partial y}\left[\frac{\partial \gamma_{yz}}{\partial x} - \frac{\partial \gamma_{zx}}{\partial y} + \frac{\partial \gamma_{xy}}{\partial z}\right] &= 2\frac{\partial^2 \epsilon_y}{\partial z \partial x} \qquad (2.10) \\
\frac{\partial}{\partial z}\left[\frac{\partial \gamma_{zx}}{\partial y} - \frac{\partial \gamma_{xy}}{\partial z} + \frac{\partial \gamma_{yz}}{\partial x}\right] &= 2\frac{\partial^2 \epsilon_z}{\partial x \partial y}
\end{aligned}
$$

Substituting Equation (2.5) into Equations (2.8) and (2.9) the Beltrami-Michell compatibility equations are obtained as follow. Let $\Theta = \sigma_x + \sigma_y + \sigma_z$ and Δ represent the Laplacian operator, then

$$\begin{aligned}
\Delta\sigma_x + \frac{1}{1+\nu}\frac{\partial^2\Theta}{\partial x^2} &= -\frac{\nu}{1-\nu}\left(\frac{\partial X}{\partial x} + \frac{\partial Y}{\partial y} + \frac{\partial Z}{\partial z}\right) - 2\frac{\partial X}{\partial x} \\
\Delta\sigma_y + \frac{1}{1+\nu}\frac{\partial^2\Theta}{\partial y^2} &= -\frac{\nu}{1-\nu}\left(\frac{\partial X}{\partial x} + \frac{\partial Y}{\partial y} + \frac{\partial Z}{\partial z}\right) - 2\frac{\partial Y}{\partial y} \qquad (2.11) \\
\Delta\sigma_z + \frac{1}{1+\nu}\frac{\partial^2\Theta}{\partial z^2} &= -\frac{\nu}{1-\nu}\left(\frac{\partial X}{\partial x} + \frac{\partial Y}{\partial y} + \frac{\partial Z}{\partial z}\right) - 2\frac{\partial Z}{\partial z}
\end{aligned}$$

For a more extensive derivation see Volterra and Gaines (1971).

Chapter 3

FINITE ELEMENT METHOD OF STRUCTURAL ANALYSIS

Introduction

The finite element method of structural analysis extends the matrix methods of structural analysis for application on high speed digital computers. It is based on the concept of replacing a continuous structure by an equivalent mathematical model made up of structural elements having known material and geometric properties expressed in matrix form. These individual matrices are then assembled following a set of rules derived from the Theory of Elasticity, to provide the geometric and material properties and static and dynamic characteristics of the structure.

The basic aims of structural analysis is to ascertain the stresses and deformations of a structure as it is subject to varied conditions of loading, temperature changes, applied deformations, etc.. As the structures become more complex, the finite element method is the only alternative to performing an accurate structural analysis. The most common matrix methods utilized to obtain the approximate solution to linearly elastic structures are the unit load method, the unit

displacement method, and the basic variational form of the displacement method derived from minimum potential energy considerations. The concise and systematic notations of matrix algebra utilized in these methods are ideally suited for programming digital computers to solve the large number of equations generated in the analysis of highly redundant structures.

Matrix methods of structural analysis are classified as matrix force or matrix displacement methods depending on whether forces or displacements are the problem coordinates. These two methods will be presented in detail in the following sections.

One dimensional elements (rods) have been chosen here to illustrate the two methods because exact solutions exist for the one dimensional problem. In addition, they are particularly useful because they allow the various steps of the solution process to be clearly examined.

Structure Displacement

In the finite element method the structure is subdivided into separate substructures known as elements which are joined and loaded at a number of points or

nodes. Each of these nodes can have up to six components of joint displacements and rotations (u, v, w, γ, β, α) called degrees of freedom. This idealized structure is known as a mesh or mathematical model.

The notations used for the force and displacement methods used in the following sections are now introduced. [Q] will be the column vector that describes the internal forces while [S] will be a column vector describing the corresponding internal displacements. A subscript will be used to show the member and direction [Q] and [S] belong to. For example, $[Q_n]$ indicates the internal forces for the Nth degree of freedom which in turn denote a particular member and direction of the internal force. If we let $[\overline{Q}]$ and $[\overline{S}]$ denote respectively the nodal forces and nodal displacements, then the external work done and the internal strain energy may be expressed as

$$[We] = [\overline{Q}]\ [\overline{S}] \tag{3.1}$$

and

$$[Wi] = [Q]\ [S] \tag{3.2}$$

Equating [We] and [Wi] we obtain

$$[\bar{Q}]\ [\bar{S}] = [Q]\ [S] \tag{3.3}$$

For a statically determinate structure, the member forces and deflections may be expressed in terms of the external nodal loads

$$[Q] = [R]\ [\bar{Q}] \tag{3.4}$$

$$[S] = [R]\ [\bar{S}] \tag{3.5}$$

where [R] is a matrix that transforms the external forces to the elements forces. For rod and beam elements, it has the physical meaning of a rotation matrix that maps the global or structure system into the local or element coordinate systems.

$$[R] \quad = \quad \begin{bmatrix} l1 & m1 & n1 \\ l2 & m2 & n2 \\ l3 & m3 & n3 \end{bmatrix} \tag{3.6}$$

where the origins of X Y Z (global) and X' Y' Z' (local) systems are the same and l1, m1, n1, l2, m2, n2, l3, m3, n3 are the direction cosines of the X' Y' Z' axes relative to the X Y Z axes, respectively.

The Matrix Force Method

In the matrix force method of structural analysis the internal loads and external reactions are the unknowns. The correct system of loads is that which satisfies the minimum energy condition.

The flexibility coefficient constitutes a relationship between the elements displacements and forces on a structure and are calculated from the geometric and material properties of the elements. This coefficient is an expression of the reduced structure and each element is defined as follow:

$f^{k\,l}_{i\,j}$ is the displacement necessary at node i acting in the k direction required to sustain a unit force at node j in the l direction

The element flexibility coefficients can be determined from deformation geometry, virtual work, or strain energy methods. The element flexibility

matrix [F] is assembled from the corresponding element flexibility coefficients. For example, for a plane rod element with two degrees of freedom per node, i.e., two translations in the X and Y directions at each end of the element, the flexibility matrix is:

$$[F] = \begin{bmatrix} f_{11}^{xx} & f_{11}^{xy} & f_{12}^{xx} & f_{12}^{xy} \\ f_{11}^{yx} & f_{11}^{yy} & f_{12}^{yx} & f_{12}^{yy} \\ f_{21}^{xx} & f_{21}^{xy} & f_{22}^{xx} & f_{22}^{xy} \\ f_{21}^{yx} & f_{21}^{yy} & f_{22}^{yx} & f_{22}^{yy} \end{bmatrix} = \begin{bmatrix} f_{11} & f_{12} & f_{13} & f_{14} \\ f_{21} & f_{22} & f_{23} & f_{24} \\ f_{31} & f_{32} & f_{33} & f_{34} \\ f_{41} & f_{42} & f_{43} & f_{44} \end{bmatrix} \quad (3.7)$$

The matrix [F] in Equation (3.7) is shown using two different notations to identify the individual matrix components. A little experimentation will show that the use of two subscripts/superscripts to identify the element flexibility coefficients is impractical when dealing with multi-element structures or writing computer codes. For this reason, degrees of freedom

numbers are used as subscripts here in this thesis to explain the theory of the force and displacement methods of analyses and in the eventual programming of the ASTRA computer code.

The structure flexibility matrix $[F_s]$ is assembled by the addition of all applicable individual element flexibility matrices [F] that are applicable.

$$[F_s] = \sum_{i=1}^{n} [R]^{T(i)} \; [F]^{(i)} \; [R]^{(i)} \tag{3.8}$$

Here the total number of elements in the structure is n, and [R] is the force transformation matrix.

The deformation of a member having n degrees of freedom (DOF) may be expressed using the flexibility coefficients in terms of the separate influences of the set of member forces [Q].

$$\begin{bmatrix} S_1 \\ S_2 \\ S_3 \\ \cdot \\ \cdot \\ \cdot \\ S_n \end{bmatrix} = \begin{bmatrix} f_{11} & f_{12} & f_{13} & \cdot & \cdot & \cdot & f_{1n} \\ f_{21} & f_{22} & f_{23} & \cdot & \cdot & \cdot & f_{2n} \\ f_{31} & f_{32} & f_{33} & \cdot & \cdot & \cdot & f_{3n} \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ f_{n1} & f_{n2} & f_{n3} & \cdot & \cdot & \cdot & f_{nn} \end{bmatrix} \begin{bmatrix} Q_1 \\ Q_2 \\ Q_3 \\ \cdot \\ \cdot \\ \cdot \\ Q_n \end{bmatrix} \tag{3.9}$$

Equation (3.9) can be more concisely expressed in the local or member coordinate system as

$$[S] = [F]\ [Q] \tag{3.10}$$

and in the structure or global coordinate systems

$$[R]\ [\bar{S}] = [F]\ [R]\ [\bar{Q}] \tag{3.11}$$

The relationship between the nodal displacements $[\bar{S}]$ and the external forces $[\bar{Q}]$ may be derived by expanding Equation (3.11) to include more than one element as follows:

$$[\bar{S}] = [R]^T [F]\ [R]\ [\bar{Q}] \tag{3.12}$$

$$[\bar{S}] = [Fs]\ [\bar{Q}] \tag{3.13}$$

where [Fs] is the structure flexibility matrix and can be found using Equation (3.8).

For statically determinate structures, Equation (3.13) gives a direct solution for all the nodal displacements in terms of the external forces acting at the nodes.

If the structure is statistically indeterminate, then the work performed by the redundant forces must be included and the compatibility conditions invoked. For these cases, Equation (3.13) becomes

$$[\bar{S}] = [F]\ [\bar{Q}] - [\bar{S}_r] \tag{3.14}$$

where $[\bar{S}r]$ are the displacements of the primary structure at the points of redundancy. The conditions of compatibility state the the displacements at all the releases of redundant points caused by the applied loads and redundant forces must be made to vanish.

Equation (3.14) may be expressed in matrix form as

$$\begin{bmatrix} \bar{S}_s \\ \hline \bar{S}_r \end{bmatrix} = \left[\begin{array}{c|c} F_{ss} & F_{sr} \\ \hline F_{rs} & F_{rr} \end{array}\right] \begin{bmatrix} \bar{Q}_s \\ \hline \bar{Q}_r \end{bmatrix} \tag{3.15}$$

where $[\bar{Q}_s]$, $[\bar{S}_s]$ stand for the structure and $[\bar{Q}_r]$, $[\bar{S}_r]$ for the redundant forces and deflections. The compatibility condition is expressed as

$$[F_{rs}]\ [\bar{Q}_s] + [F_{rr}]\ [\bar{Q}_r] = [\bar{S}_r]$$

however, if the redundant coordinates are fixed in space, $[\bar{S}_r] = 0$ and

$$[F_{rs}]\ [\bar{Q}_s] + [F_{rr}]\ [\bar{Q}_r] = 0 \tag{3.17}$$

from which we obtain

$$[\bar{Q}_r] = -[F_{rr}]^{-1}\ [F_{rs}]\ [\bar{Q}_s] \tag{3.18}$$

which describes the solution for the redundant loads.

The solution procedure for redundant structures then is:

1. Define the external loads $[\bar{Q}]$ and specify the redundancies.

2. Calculate the force transformation matrix [R] and the element transformation matrix [F] for all the elements.

3. Assemble the structure flexibility matrix $[F_s]$.

4. Solve for the redundant forces $[\bar{Q}_r]$.

5. Calculate the nodal displacements $[\bar{S}_s]$.

6. Solve for the member forces [Q].

For a more extensive derivation of the force method see Yan-Yu Hsieh (1970).

The Displacement Method

In the displacement method of structural analysis, the structures displacements are the basic unknowns. A set of equations of equilibrium equal to the degree of kinematic indeterminacy (number of unknown diplacements) has to be solved in order to determine these displacements.

In this method, the compatibility conditions are first satisfied by correlating the external nodal displacements to the displacements of its members. Then a force-displacements relationship is established between the member's end forces and deformations. Finally, the equilibrium equations are used to calculate the nodal displacements, member forces, and reactions of the structure.

The compatibility conditions used in the displacement method express the condition that the member deformations [S] must be consistently related to the nodal displacements $[\bar{S}]$. Let R_{ij} represent the value of the member deformations S_i caused by a unit nodal displacement $\bar{S}_j$. The value of the total member deformations [S] caused by all the nodal displacements $[\bar{S}]$ may be written as

$$\begin{bmatrix} S_1 \\ S_2 \\ S_3 \\ \cdot \\ \cdot \\ \cdot \\ S_n \end{bmatrix} = \begin{bmatrix} R_{11} & R_{12} & R_{13} & \cdots & R_{1n} \\ R_{21} & R_{22} & R_{23} & \cdots & R_{2n} \\ R_{31} & R_{32} & R_{33} & \cdots & R_{3n} \\ \cdot & \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdot & \cdots & \cdot \\ R_{n1} & R_{n2} & R_{n3} & \cdots & R_{nn} \end{bmatrix} \begin{bmatrix} \bar{S}_1 \\ \bar{S}_2 \\ \bar{S}_3 \\ \cdot \\ \cdot \\ \cdot \\ \bar{S}_n \end{bmatrix} \qquad (3.19)$$

or

$$[S] = [R]\ [\bar{S}] \qquad (3.5)$$

and similarly we have

$$[Q] = [R]\ [\bar{Q}] \qquad (3.4)$$

where $[Q]$ is the vector of element forces and $[\bar{Q}]$ the vector of nodal forces. The matrix $[R]$ is the displacement transformation matrix and relates the internal member deformations to the external nodal displacements. For beam and rod elements it can be physically represented as a geometric transformation of coordinates from the structure coordinate system to the element coordinate system as mentioned previously.

For all structural elements a direct relationship exists between the forces applied at the boundaries and the boundary displacements. This relationship can be expressed in matrix notation as

$$[Q] = [K]\ [S] \tag{3.20}$$

where [Q] is the vector of element loads, [S] is a vector of element displacements and [K] is the element stiffness and is defined as follows:

K_{ij}^{ml} is the force at node i acting in the m direction required to sustain a unit displacement at node j in the l direction

and, for example, in the case of a two dimensional rod

$$[K] = \begin{bmatrix} k_{11}^{xx} & k_{11}^{xy} & k_{12}^{xx} & k_{12}^{xy} \\ k_{11}^{yx} & k_{11}^{yy} & k_{12}^{yx} & k_{12}^{yy} \\ k_{21}^{xx} & k_{21}^{xy} & k_{22}^{xx} & k_{22}^{xy} \\ k_{21}^{yx} & k_{21}^{yy} & k_{22}^{yx} & k_{22}^{yy} \end{bmatrix} = \begin{bmatrix} k_{11} & k_{12} & k_{13} & k_{14} \\ k_{21} & k_{22} & k_{23} & k_{24} \\ k_{31} & k_{32} & k_{33} & k_{34} \\ k_{41} & k_{42} & k_{43} & k_{44} \end{bmatrix} \tag{3.21}$$

The element stiffness matrix [K] and the structure stiffness matrix [Ks] are assembled by the addition of

all the individual stiffness coefficients and element's stiffness matrices respectively in a manner similar to that employed to asssemble the element and structure flexibility matrices.

For a structure composed of one element having n degrees of freedom (DOF) we have

$$\begin{bmatrix} Q_1 \\ Q_2 \\ Q_3 \\ \cdot \\ \cdot \\ \cdot \\ Q_n \end{bmatrix} = \begin{bmatrix} k_{11} & k_{12} & k_{13} & \cdot & \cdot & \cdot & k_{1n} \\ k_{21} & k_{22} & k_{23} & \cdot & \cdot & \cdot & k_{2n} \\ k_{31} & k_{32} & k_{33} & \cdot & \cdot & \cdot & k_{3n} \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ k_{n1} & k_{n2} & k_{n3} & \cdot & \cdot & \cdot & k_{nn} \end{bmatrix} \begin{bmatrix} S_1 \\ S_2 \\ S_3 \\ \cdot \\ \cdot \\ \cdot \\ S_n \end{bmatrix} \tag{3.22}$$

which expressed in abbreviated form becomes

$$[Q] = [k]\ [S] \tag{3.20}$$

For an idealized structure composed of multiple members, the nodal forces and nodal displacements can be expressed in terms of the element forces and displacements by substitution of Equation (3.4) and (3.5) into Equation (3.20) to obtain

$$[R]\ [\bar{Q}] = [K]\ [R]\ [\bar{S}] \tag{3.23}$$

or

$$[\bar{Q}] = [R]^{T}\ [K]\ [R]\ [\bar{S}]$$

which reduce to

$$[\bar{Q}] = [Ks]\ [\bar{S}] \tag{3.24}$$

where [Ks] is the structure stiffness and is assembled from the sum of all the individual element stiffness matrices expressed in the global or structure coordinate system.

$$[Ks] = \sum_{i=1}^{n} [R]^{T(i)}\ [K]^{(i)}\ [R]^{(i)} \tag{3.25}$$

The solution procedure of the displacement method is straight forward and can be summarized as follows:

1. Compile the basic data of the structure (material properties, geometry, load conditions, etc.)

2. Construct the idealized structure; define the location of nodes, element orientation (topology), element material and geometric characteristics.

3. Calculate the element stiffness [K].

4. Calculate the element transformation matrices [R].

5. Calculate the structure stiffness

$$[Ks] = \sum_{i=1}^{n} [R]^{T(i)} [K]^{(i)} [R]^{(i)} \qquad (3.25)$$

6. Eliminate degrees of freedom (DOF) not used to establish the reduced structures stiffness [Kr], loads $[\overline{Q}_r]$, and displacement $[\overline{S}_r]$ matrices. Loads and displacement matrices for the redundancies are related by

$$[\overline{Q}_r] = [K_r] [\overline{S}_r] \qquad (3.26)$$

7. Partition all matrices. Partitioning of matrices is used to lessen the work in matrix multiplication and inversion.

$$\left[\bar{Q}_r\right] = \begin{bmatrix} \bar{Q}_\alpha \\ \bar{Q}_\beta \end{bmatrix} \tag{3.27}$$

$$\left[K_r\right] = \begin{bmatrix} K_{\alpha\alpha} & K_{\alpha\beta} \\ K_{\beta\alpha} & K_{\beta\beta} \end{bmatrix} \tag{3.28}$$

$$\left[\bar{S}_r\right] = \begin{bmatrix} \bar{S}_\alpha \\ \bar{S}_\beta \end{bmatrix} \tag{3.29}$$

Substituting Equation (3.27) through (3.29) into Equation (3.26)

$$\begin{bmatrix} \bar{Q}_\alpha \\ \bar{Q}_\beta \end{bmatrix} = \begin{bmatrix} K_{\alpha\alpha} & K_{\alpha\beta} \\ K_{\beta\alpha} & K_{\beta\beta} \end{bmatrix} \begin{bmatrix} \bar{S}_\alpha \\ \bar{S}_\beta \end{bmatrix} \tag{3.30}$$

Note that

$$[K_{\alpha\beta}] = [K_{\beta\alpha}]^T$$

and $[\overline{S}_{\alpha}]$ is the vector representing the degrees of freedom which are free to move or displace. $[\overline{S}_{\beta}]$ is the vector of degrees of freedom that are fixed in space. If the loads are expressed as applied forces and not as given displacement $[\overline{S}_{\beta}] = 0$. Similarly, $[\overline{Q}_{\alpha}]$ and $[\overline{Q}_{\beta}]$ denote the nodal loads and reactions, respectively.

8. Calculate the unknown displacements

$$[K_{\alpha\alpha}]^{-1} [\overline{Q}_{\alpha}] = [\overline{S}_{\alpha}] \qquad (3.31)$$

9. Calculate the reactions

$$[\overline{Q}_{\beta}] = [K_{\beta\alpha}] [\overline{S}_{\alpha}] \qquad (3.32)$$

10. Calculate the element forces and stresses

$$[Q]^{(i)} = [K]^{(i)} [R]^{(i)} [S_{\alpha}] \qquad (3.33)$$

Examples of structural problems using the displacement method are shown in Appendix E.

Comparison of The Force and Displacement Methods

When comparing the force and the displacement methods of analysis, it must be emphasized that the same input information and element properties can be used regardless of the method to be employed in the analysis. Both methods lead to the same theoretical result, one being the inverse of the other. The computational path, however, leading to the calculation of displacements is different in each method. Because of possible ill-conditioning of the equations, different rounding-off errors, and the actual numerical analysis, the numerical results may differ for the two methods.

If the matrix operations for both methods are compared, it is found that in the force method, through the use of the Jordanian elimination technique (Przemieniecki, 1968), the sequence of matrix operations is considerably more complicated than that for the displacement method. When writing computer programs for the solution of structural problems, except for structures which involve many joint displacements and few redundants, displacement methods or a combination of displacement and force methods are usually preferred. The displacement method uses the same procedures for analyzing statically determinate and statically

indeterminate structures, operates directly on the complete structure, and does not require a reduced structure as does the force method. In addition, in the displacement method each unit load is applied to each degree of freedom while all others are held zero. The equations also tend to be better conditioned, i.e., the largest terms lie on the main diagonal. In the force method a well conditioned flexibility matrix depends upon a good choice of redundants. Finally, for vibration problems the stiffness matrix [Ks] of the displacement method is determined directly and is better conditioned and easier to work with than the flexibility matrix [Fs] of the force method.

Chapter 4

THE COMPUTER PROGRAM "ASTRA"

Introduction

The computer program "Analyzing Structures with the APPLE" or "ASTRA" for short, is a finite element program based on the displacement method using the APPLE II or a similar microcomputer. The intented range of applications of the program extends to the solution of static structural problems in a plane, and in space, using a combination of beam and truss members with concentrated point loads and moments at the joints.

The program has been written in such a way so as to allow conversion to microcomputer systems other than the APPLE II or to more advanced programming languages. Eventual expansion to include more advanced element types, and increases in the number of elements and nodes that it can handle has also been facilitated. This has been accomplished by the use of subroutines and comment statement. Although the use of comments involves a substantial use of available RAM, their inclusion in the program makes it much easier to understand and to follow the course of the matrix operations and use of the subroutines.

When working with microcomputers, the programmer must make a decision as to how to best employ the amount of RAM and disk storage available. In tailoring ASTRA to the APPLE computer system, it was found that although program running time is critical to the efficient use of the computer, the limiting factor was the amount of memory available. Thus, the heart of of the ASTRA program is the efficient use of available disk storage, and the use of special purpose subroutines to speed up matrix operations. This is accomplished primarily in the calculation, storage, and retrieval of the structure stiffness matrix [Ks] and the partitioned stiffness matrices $[K_{\alpha\alpha}]$ and $[K_{\alpha\beta}]$. Only the lower half of the bandwidth below the matrix diagonal is stored. Although this feature increases the running time and complexity of the program, it allows ASTRA to be run on a microcomputer by using the disk drive as the main storage device.

While developing ASTRA, maximum thought was given to making the computer program "user oriented", i.e., a program that would be easy to use by anyone having some background in structural analysis with a minimum amount of training. This was accomplished by the use of the pre-processor "IASTRA" or "Input to

ASTRA." All that the user has to do to prepare or change input information required to run ASTRA is to answer the questions formulated by the computer. It is felt that this approach eliminates many of the problems associated with running a computer program and thus frees the user to concentrate on the analysis of the structure.

The first version of the computer program ASTRA is the result of two years of research and experience with other finite element computer programs commercially available on large computers. It is hoped that it will become a flexible and efficient tool for the analysis of structures. The program presently contains the following types of elements:

a) Three-dimensional truss element (rod)

b) Three-dimensional bar element (beam)

The structure to be statically analyzed may be composed of a combination of a number of these elements.

The capacity of the program depends on the type and number of storage devices used, computer language, and time available for the analysis. Today, it is felt that a comfortable number of elements capable of being handled by the program is twenty-five. This is limited by the use of an APPLE II equipped with 48K of RAM and

116K mini-floppy disk. However, should a hard disk drive (10 megabytes) and an APPLE III with Pascal and 124K RAM become available, the capacity of the program could be tremendously improved and the running time reduced by using matrix bandwidth storage techniques.

The purpose of this section is to present the concept and the theory of the program ASTRA, and show the formulation of the element stiffness and transformation matrices and the solution procedures.

Method of Analysis

ASTRA is a finite element program for the solution of static problems based on the displacement method of structural analysis. The finite element method and general equations which govern the equilibrium of the system are given in Chapter 3. However, for completeness, the equations and solutions procedures are reviewed in the following discussion.

The relationship between the forces applied at the boundary and the displacements in a structural element can be expressed as follows:

$$[Q] = [K]\ [S] \tag{3.20}$$

where [Q] is the vector of elements loads, [K] is the element stiffness and [S] is the vector of element displacements. The member deformations [S] and forces [Q] can be related to the nodal displacements $[\overline{S}]$ and nodal forces $[\overline{Q}]$. This can be expressed in matrix notation as

$$[Q] = [R]\ [\overline{Q}] \tag{3.4}$$

and

$$[S] = [R]\ [\overline{S}] \tag{3.5}$$

The same relation can be defined globally for the complete structure.

$$[\overline{Q}] = [Ks]\ [\overline{S}] \tag{3.24}$$

where [Ks] is the structure stiffness matrix as defined by Equation (3.25).

$$[Ks] = \sum_{i=1}^{n} [R]^{T(i)}\ [K]^{(i)}\ [R]^{(i)} \tag{3.25}$$

After the model is constructed, the element transformation and stiffness matrices are calculated and the structure stiffness matrix is assembled. Then, as expressed in Equation (3.26), the redundant displacements are eliminated to establish the reduced structure stiffness, displacements and loads

$$[\overline{Q}r] = [Kr]\ [\overline{S}r] \tag{3.26}$$

and consequently partitioned per Equation (3.27) through (3.29).

$$\left[\ \overline{Q}\ \right] = \begin{bmatrix} \overline{Q}_\alpha \\ \overline{Q}_\beta \end{bmatrix} \tag{3.27}$$

$$\left[\ K_s\ \right] = \begin{bmatrix} K_{\alpha\alpha} & K_{\alpha\beta} \\ K_{\beta\alpha} & K_{\beta\beta} \end{bmatrix} \tag{3.28}$$

$$\left[\ \overline{S}\ \right] = \begin{bmatrix} \overline{S}_\alpha \\ \overline{S}_\beta \end{bmatrix} \tag{3.29}$$

Substituting Equation (3.27) through (3.29) into Equation (3.26) yields

$$\begin{bmatrix} \bar{Q}_\alpha \\ \bar{Q}_\beta \end{bmatrix} = \begin{bmatrix} K_{\alpha\alpha} & K_{\alpha\beta} \\ K_{\beta\alpha} & K_{\beta\beta} \end{bmatrix} \begin{bmatrix} \bar{S}_\alpha \\ \bar{S}_\beta \end{bmatrix} \tag{3.30}$$

If there are no induced displacement in the structure $[S_\beta] = 0$ and the unknown displacements, reactions and element forces are calculated

$$[K_{\alpha\alpha}]^{-1} \quad [\bar{Q}_\alpha] = [\bar{S}_\alpha] \tag{3.31}$$

$$[\bar{Q}_\beta] = [K_{\beta\alpha}] \quad [\bar{S}_\alpha] \tag{3.32}$$

$$[Q]^{(i)} = [K]^{(i)} \quad [R]^{(i)} \quad [\bar{S}_\alpha] \tag{3.33}$$

For a more complete derivation of the displacement methods see Hsieh (1970).

Structural Elements

The present version of the computer program ASTRA contains two structural elements. These are three-dimensional rods and beam elements. These two element types were selected because their stiffness and displacement transformation matrices are easy to calculate and exact solutions exist to check the final

results. In addition, their use enables the various steps of the solution procedure to be examined and it simplifies the programmer's task of debugging the computer program.

The element description, characteristics, and calculation of the element stiffness and displacement transformation matrices are presented in the following sections. For a more detailed discussion see Hsieh (1970), and Przemieniecki (1968).

The rod element shown in Figure 2 is a simple tension-compression bar element pin jointed at the ends and with uniform cross-sectional area. The beam element illustrated in Figure 3 is assumed to be a straight bar of uniform cross-section capable of resisting axial forces, bending moments about the two principal axes in the plane of its cross-section, and torsion about its centroidal axes.

The stiffness matrices for these elements are of order 12 x 12. This matrix size was chosen as each node has six degrees of freedom (DOF) and there are two nodes per element. Also, it is much easier to eliminate redundant degrees of freedom in the structure stiffness matrix [Ks], as shown in Equation (3.26), than to keep track of which node has how many degrees of freedom.

If the local axes are chosen to coincide with the principal axes of the cross-section, it is possible to construct the 12 x 12 stiffness matrix from 2 x 2 and 4 x 4 submatrices. The force displacement relationships for the uniform beam element will now be derived directly from the differential equations for beam displacements. The stiffness coefficients thus obtained will be exact within the limits of the assumptions in the general theory of beams subjected to loads.

The differential equations for the axial displacements of the beam shown in Figure 4 have the boundary conditions that at X = 0 the displacement is S1 while at X = L, $S_7 = 0$.

$$Q_1 = -\left(\frac{dS}{dx}\right)\left(EA\right) \tag{4.1}$$

which can be integrated directly. For the above boundary conditions

$$Q_1 = (EA/L)S_1 \tag{4.2}$$

from the equations of equilibrium in the X direction it follows that

$$Q_1 = -Q_7, \tag{4.3}$$

then

$$k_{1,1} = k_{7,7} = EA/L \qquad (4.4)$$

and employing equilibrium conditions

$$k_{1,7} = k_{7,1} = -EA/L \qquad (4.5)$$

The differential equation for the twist on the beam shown in Figure 5 is

$$Q_4 = -GJ\,\frac{d\theta}{dx} \qquad (4.6)$$

where GJ is the torsional stiffness of the beam cross section. By integrating Equation (4.6), Equation (4.7) is obtained.

$$Q_4 x = -GJ\,\theta + C_1 \qquad (4.7)$$

With the boundary conditions $\theta = 0$ at $X = L$, C_1 is calculated

$$C_1 = Q_4 L \qquad (4.8)$$

and since $\theta = S_4$ from Equations (4.7) and (4.8) we obtain

$$Q_4 = (GJ/L)S_4 \tag{4.9}$$

For the twisting moments we have

$$Q_{10} = -Q_4 \tag{4.10}$$

Hence

$$k_{4,\,4} = k_{10,\,10} = GJ/L \tag{4.11}$$

and

$$k_{4,\,10} = k_{10,\,4} = -GJ/L \tag{4.12}$$

The lateral deflection S on the beam shown in Figure 6, subjected to shearing forces and associated moments, is given by

$$S = Sb + Ss \tag{4.13}$$

where Sb is the lateral deflection due to bending strains and Ss is the additional deflections due to shearing strains. These deflections can be expressed respectively as:

$$EI_z(dSb/dx) = Q_2X - Q_6 \qquad (4.14)$$

and

$$dSs/dX = -Q_2/GAs \qquad (4.15)$$

Integration of Equations (4.14) and (4.15) gives

$$EI_z S = \frac{Q_2 X^3}{6} - \frac{Q_6 X^2}{2} - \frac{Q_2 EI_z X}{GA_s} + C_1X + C_2 \qquad (4.16)$$

where As represents the effective beam shear area. The constants of integration C_1 and C_2 can be found from the boundary conditions shown in Figure 6.

At X = 0 and X = L

$$\frac{dS}{dx} = \frac{dSs}{dx} \qquad (4.17)$$

and at X = L

$$S = 0$$

Equation (4.16) then becomes

$$E I_z S = \left(\frac{Q_2 X^3}{6}\right) - \left(\frac{Q_6 X^2}{2}\right) -$$

$$\left(\frac{Q_2 \Phi X L^2}{12}\right) + \left(1 + \Phi\right) \left(\frac{Q_2 L^3}{12}\right) \tag{4.18}$$

where

$$Q_6 = \frac{Q_2 L}{2} \qquad (4.19)$$

and

$$\Phi = \frac{12 E I_z}{G A_s L^2} \qquad (4.20)$$

The boundary conditions for the built-in end are taken as $\theta = 0$, i.e., the slope due to bending deformation is equal to zero. The remaining forces acting on the beam can be determined from equilibrium.

$$Q_8 = -Q_2 \qquad (4.21)$$

and

$$Q_{12} = -Q_6 + Q_2 L \qquad (4.22)$$

At $X = 0$ and $S = S_2$, we obtain from Equation (4.18)

$$S_2 = (1 + \Phi)\frac{Q_2 L^3}{12 E I_z} \qquad (4.23)$$

By using Equations (4.19) and (4.21) through (4.23) and by the use of the conditions of symmetry, or the differential equations for the beam deflections, it can be shown that

$$k_{2,\,2} = K_{8,\,8} = -k_{8,\,2} = (12EI)/(1+\Phi)L^3 \tag{4.24}$$

and

$$K_{6,\,2} = k_{12,\,2} = k_{12,\,8} = (6EI)/(1+\Phi)L^2 \tag{4.25}$$

A beam subjected to bending moments and associated shears is shown in Figure 7. In order to calculate the deflections Equation (4.16) is evaluated with a new set of boundary conditions. Let S = 0 at X = 0 and X = L and at X = L

$$\frac{dS}{dx} = \frac{dS_s}{dx} \tag{4.26}$$

then Equation (4.16) becomes

$$EI_zS = \frac{Q_2}{6}\left(X^3 - L^2X\right) + \frac{Q_6}{2}\left(LX - X^2\right) \tag{4.27}$$

and

$$Q_2 = \frac{6\ Q_6}{\left(4 + \Phi\right) L} \qquad (4.28)$$

The remaining forces acting on the beam can be determined from Equations (4.21) and (4.22). At $x = 0$

$$\frac{dS_b}{dx} = \frac{dS}{dx} - \frac{dS_s}{dx} = S_6 \qquad (4.29)$$

and S_6 becomes

$$S_6 = \frac{Q_6 (\ 1 + \Phi\ ) L}{EI_z (\ 4 + \Phi\ )} \qquad (4.30)$$

Thus, the stiffness coefficients are obtained from Equations (4.21), (4.22), (4.28) and (4.30).

$$k_{6,6} = k_{12,12} = \frac{(4 + \Phi) EI_z}{(\ 1 + \Phi) L} \qquad (4.31)$$

$$k_{8,6} = \frac{-6EI_z}{(1+\Phi)L^2} \tag{4.32}$$

$$k_{12,6} = \frac{(2-\Phi)EI_z}{(1+\Phi)L} \tag{4.33}$$

The stiffness coefficients associated with the deflections S_3 and S_5 can be derived following a similar approach.

The element stiffness matrix [K] is assembled from the individual stiffness coefficients k and for a beam element are shown in Table 1. If the shear parameter Φ representing shear deformation is taken as zero in Table 1, Table 2 is produced. The member stiffness matrix for a rod element is similarly constructed by assembling the individual stiffness coefficients, Equations (4.1) through (4.5), as shown in Table 3.

TRANSFORMATION MATRICES

The member stiffness matrix [K] must be converted from the member to the structure coordinate system in order to add the contribution of each member stiffness to the structure stiffness matrix [Ks] as shown in Equation (3.25),

$$[Ks] = \sum_{i=1}^{n} [R]^{T(i)} [K]^{(i)} [R]^{(i)} \qquad (3.25)$$

Consider the body shown in Figure 8. Two sets of orthogonal axes with origins at 0 are shown. The Xm, Ym, and Zm axes will be taken as a set of member oriented axes and referred to as the member or local coordinate system. The Xs, Ys, and Zs axes are assumed to be parallel to a set of structure reference axes identified as the global or structure coordinate system. Let λij represent the direction cosines of the Xm, Ym, and Zm axes with respect to the Xs, Ys, and Zs axes, i.e., the cosines of the angles between the member and the structure axes. Let [Am] be the set of coordinates, written in matrix format, for a point expressed in the member coordinate system and [As] the set of coordinates for the same point expressed in the

structure coordinate system. The relationship between [Am] and [As] can be expressed as follows:

$$[Am] = \begin{bmatrix} Xm \\ Ym \\ Zm \end{bmatrix} \tag{4.34}$$

$$[As] = \begin{bmatrix} Xs \\ Ys \\ Zs \end{bmatrix} \tag{4.35}$$

and

$$[Am] = [R]\ [As] \tag{4.36}$$

where [R] is a transformation matrix of the form

$$[R] = \begin{bmatrix} \lambda_{11} & \lambda_{12} & \lambda_{13} \\ \lambda_{21} & \lambda_{22} & \lambda_{23} \\ \lambda_{31} & \lambda_{32} & \lambda_{33} \end{bmatrix} \tag{4.37}$$

and λ_{ij} represents the direction cosines with the first subscript referring to the Xm, Ym, and Zm axes and the second subscript referring to the Xs, Ys, and Zs axes.

The rotation matrix [R] for a space rod is of indefinite form because the orientation of its principal axes is unspecified. Let the Xm axis of the member coordinates be chosen to coincide with the longitudinal axis of the member and the member Zm axis be taken along the global Zs axis in order to define its position in all cases, including that of a vertical member. The rotation matrix [R] for the space rod then takes the general form shown in Equation (4.38), where Cx, Cy, and Cz are the direction cosines of the member axis with respect to the global Xs axis.

$$[R] = \begin{bmatrix} C_x & C_y & C_z \\ \lambda_{21} & \lambda_{22} & \lambda_{23} \\ \lambda_{31} & \lambda_{32} & \lambda_{33} \end{bmatrix} \qquad (4.38)$$

The direction cosines for the last two rows of [R] in Equation (4.38) can be found directly from Figure 9 by geometric considerations. Alternatively,

the transformation may be considered to take place in two steps; A rotation about the Ys axis through an angle β and a rotation through an angle γ about the Zs axis. The matrix $[R_\beta]$ is composed of the direction cosines of the intermediate β axes with respect to the structure axes.

$$[R_\beta] = \begin{bmatrix} \dfrac{C_x}{\sqrt{C_x^2+C_z^2}} & 0 & \dfrac{C_z}{\sqrt{C_x^2+C_z^2}} \\ 0 & 1 & 0 \\ \dfrac{-C_z}{\sqrt{C_x^2+C_z^2}} & 0 & \dfrac{C_x}{\sqrt{C_x^2+C_z^2}} \end{bmatrix} \tag{4.39}$$

and the matrix $[R_\gamma]$ consists of the direction cosines of the member axes with respect to the β axes.

$$[R_\gamma] = \begin{bmatrix} \sqrt{C_X^2 + C_Z^2} & C_Y & 0 \\ -C_Y & \sqrt{C_X^2 + C_Z^2} & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.40}$$

The rotation matrix [R] can now be obtained as the product of the matrices $[R_\gamma]$ and $[R_\beta]$ as follows:

$$[R] = [R_\gamma]\,[R_\beta] \tag{4.41}$$

and

$$[R] = \begin{bmatrix} C_X & C_Y & C_Z \\ \dfrac{-C_X C_Y}{\sqrt{C_X^2 + C_Z^2}} & \sqrt{C_X^2 + C_Z^2} & \dfrac{-C_Y C_Z}{\sqrt{C_X^2 + C_Z^2}} \\ \dfrac{-C_Z}{\sqrt{C_X^2 + C_Z^2}} & 0 & \dfrac{C_X}{\sqrt{C_X^2 + C_Z^2}} \end{bmatrix} \tag{4.42}$$

with Cx, Cy, Cz being the direction cosines of the member axis. The transformation matrix [R] shown in Equation (4.42) is valid for all positions of the member except when it is vertical. When the member is vertical

the direction cosines of the member axes with respect to the structure can be determined by inspection and [R] becomes

$$[R] = \begin{bmatrix} 0 & Cy & 0 \\ -Cy & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.43}$$

In this matrix Cy = 1 when the member points "up" and equals to -1 when the member points "down".

The transformation matrix for a space beam may, under special circumstances, be the same as that for a space rod. However, it generally has its principal axes Ym and Zm in skew directions and a more complicated transformation matrix need be generated. A space beam with the rotations γ , β , and α is shown in Figure 10. The rotation through the angle α is further illustrated in Figure 11 and requires the introduction of the rotation matrix $[R_\alpha]$ in which the elements are the directions cosines between the member axes Xm, Ym, and Zm and the axes:

$$[R_\alpha] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & \sin\alpha \\ 0 & -\sin\alpha & \cos\alpha \end{bmatrix} \tag{4.44}$$

The required rotation matrix [R] for the space beam element can be obtained as the product of $[R_\gamma]$, $[R_\beta]$, and $[R_\alpha]$:

$$[R] = [R_\gamma]\,[R_\beta]\,[R_\alpha] \tag{4.45}$$

and

$$\left[R\right] = \begin{bmatrix} C_X & C_Y & C_Z \\ \dfrac{-C_X C_Y \cos\alpha - C_Z \sin\alpha}{\sqrt{C_X^2 + C_Z^2}} & \sqrt{C_X^2 + C_Z^2}\cos\alpha & \dfrac{-C_Y C_Z \cos\alpha + C_X \sin\alpha}{\sqrt{C_X^2 + C_Z^2}} \\ \dfrac{C_X C_Y \sin\alpha - C_Z \cos\alpha}{\sqrt{C_X^2 + C_Z^2}} & -\sqrt{C_X^2 + C_Z^2}\sin\alpha & \dfrac{C_Y C_Z \sin\alpha + C_X \cos\alpha}{\sqrt{C_X^2 + C_Z^2}} \end{bmatrix} \tag{4.46}$$

where Cx, Cy, Cz are the direction cosines of the member axes with respect to the global Xs axis and α is the angle between the $Y\gamma$ and Ym axis.

For the special case of a vertical member there is a rotation about the Zm axis through the angle γ which may be either 90 or 270. The second rotation is about the Xm axis and through an angle α . The rotation matrix for the vertical beam member can thus be obtained by inspection and consists of the direction cosines of the member axes Xm, Ym and Zm with respect to the structure axes Xs, Ys, and Zs.

$$[R] = \begin{bmatrix} 0 & Cy & 0 \\ -Cy\ \cos\alpha & 0 & \sin\alpha \\ Cy\ \sin\alpha & 0 & \cos\alpha \end{bmatrix} \qquad (4.47)$$

Again, in this case, CY = +1 when the member points "up", and CY = -1 when the member point "down". Equation (4.47) is valid for all possible vertical orientations of the member.

The orientation of a particular member may be such that the angle α may not be readily available and thus a technique for calculating the angle of rotation must be developed. A suitable method for specifying is to use the coordinates of a point that lies in one of the principal planes (Ym-Xm), but not on the axis of the member itself. This point and the Xm axis will define

without ambiguity a plane in space. All that is necessary now is to obtain an expression for the angle of rotation α , shown in Figures 10 and 11, in terms of the coordinates of the given points and the coordinates of the ends of the members. Let Xps, Yps, and Zps be the coordinates of point P with respect to the structure axis. Xp, Yp, and Zp are the point coordinates of P, as shown in Figure 11, and Xj, Yj, and Zj represent the coordinates at the end of the member (node j), then

$$\begin{aligned} Xps &= Xp - Xj \\ Yps &= Yp - Yj \\ Xps &= Zp - Zj \end{aligned} \tag{4.48}$$

The coordinates of P with respect to the axes Xs, Ys, Zs (principal axes in the structure or global system) can be obtained by

$$\begin{bmatrix} Xp\gamma \\ Yp\gamma \\ Zp\gamma \end{bmatrix} = [R_{\gamma}]\ [R_{\beta}] \begin{bmatrix} Xps \\ Yps \\ Zps \end{bmatrix} \tag{4.49}$$

Then, from the geometry of the beam shown in Figure 11, the following expressions for the sine and cosine of the rotation angle α in terms of the coordinates of the given point and those of the ends of the member are:

$$\text{Sin}\,\alpha = \frac{Z_{p\gamma}}{\sqrt{Y_{p\gamma}^2 + Z_{p\gamma}^2}} \tag{4.50}$$

and

$$\cos\alpha = \frac{Y_{p\gamma}}{\sqrt{Y_{p\gamma}^2 + Z_{p\gamma}^2}} \tag{4.51}$$

The preceding discussion dealt with non-vertical members. In the case of the vertical member, it is possible to calculate $\cos\alpha$ and $\sin\alpha$ directly from the coordinates of the point P

$$\sin\alpha = \frac{Z_{ps}}{\sqrt{X_{ps}^2 + Z_{ps}^2}} \tag{4.52}$$

and

$$\cos\alpha = \frac{-X_{ps}}{\sqrt{X_{ps}^2+Z_{ps}^2}}\ Cy \qquad (4.53)$$

Equations(4.47), (4.52), and (4.53) are general formulas valid regardless of whether the member is pointing "up or "down".

It can be summarized that the rotation [R α] for a space beam member can be generated using either the rotation angle α directly, or from the coordinates of a point P that helps to identify a principal plane (in ASTRA the Xm-Ym plane). If the angle α is zero, [R] reduces to one of the expressions generated for a space rod member, Equations (4.42) and (4.43). If α is not zero then [R] takes the form of Equation (4.46) or (4.47).

Chapter 5

A PROGRAMMERS GUIDE TO "ASTRA"

With the exception of very elementary problems, the application of the finite element method necessarily implies a considerable amount of computation, thus, the use of a digital computer is implicit. In this chapter the computer program ASTRA will be described. Its sequence of computer operations, handling and storage of matrices, and the special purpose subroutines are listed. A general program flowchart can be found in Appendix D.

The Program

In order to circumvent the memory size limitation of the APPLE II microcomputer (48k bytes), the program ASTRA was divided into a number of subprograms and subroutines. The difference between subprograms and subroutines as defined is that a subroutine is a program called for from a subprogram, while a subprogram is a complete computer program that is a part of the main program. ASTRA has been divided into 4 separate subprograms called ASTRA-1 through -4. Each of these subprograms must be executed in sequence.

In the computer system only one floppy disk was available. This forces the user to separately load and execute each of the four ASTRA subprograms. Should another floppy disk drive or hard disk drive become available to store both the programs and the data, the programs could all be set to load and run automatically by the use of a text editor file. This file is the IBM equivalent of a CLIST type file and allows for instructions to the computer to be stored and executed as though it were a regular computer program. Should Pascal, a higher level computer language, be used instead of the Applesoft Basic interpreted language used on the APPLE II, considerable amount of core storage could be saved and program execution would be speeded considerably.

The program ASTRA assigns six degrees of freedoms (DOF) to each nodal point. These degrees of freedom, 1 through 6, are equivalent to three translations and three rotations in and about the global X, Y, and Z axis respectively. In turn, the assigned degrees of freedom are not activated until the individual nodes are called for by an individual element. This feature allows for faster program execution by reducing the equilibrium equations that

must be solved for. In addition, it allows the use of dummy nodes, i.e., nodes which have been given coordinates in the program but are actually not used by any of the elements. This provides for the addition and deletion of elements and nodes without the necessity for renumbering all the nodes in the structure.

For example, consider the structure shown in Figure 13 and its corresponding model. The individual numbers shown are the node numbers and those enclosed in parenthesis are the element numbers. Let elements 1 and 2 be rod elements, and element 3 be a beam element. ASTRA would assign 18 (6 DOF x 3 nodes) degrees of freedom to the structure. Of these, only 15 degrees of freedom would be activated by the elements. Six DOF each at nodes 2 and 3 by the beam element and three DOF at node 1 by the rod elements would be activated.

In order to check whether a DOF is active or not, the user is referred to the ASTRA print out when DOFIU (degrees of freedom in use) in the computer program is displayed.

In ASTRA the high speed available memory of the APPLE II (RAM) is used primarily to store the arrays currently in use, and to provide a buffer for the solution of equations. The low speed memory provided by

the floppy disk drive is used for the storage of element properties, structure stiffness and transformation matrices, input data, and any large array. The total capacity of the program is thus limited by the amount of RAM and disk storage available while its speed of execution is governed by the total number of DOF in the structure. Because the speed of execution of ASTRA is proportional to the number of equilibrium equations to be solved, i.e., the number of active DOF in the structure, to increase the speed of execution it is recommended that all DOF not compatible with the elements connected to the nodal points be suppressed. This will reduce the number of degrees of freedom at each node and thus speed the execution of the program by decreasing the order of the structure stiffness matrix [Ks], and allowing for the analysis of larger structures in less amount of time. For example, in a plane structure in the X-Y plane, suppress all translations in the Z direction and rotations about the X axis. This will reduce the number of degrees of freedom at each node by a factor of two and the order of the matrix [Ks] by a factor of 2n, where n is the number of nodes in the structure.

Often interesting problems involving large matrices cannot be solved because they are too large and are either impossible or very expensive to invert with available computer storage. If the matrices involved are sparse matrices, there are ways of optimizing their storage so as to allow for the solution of larger problems. Sparse matrices are matrices having only a small percentage of nonzero elements. If these elements are grouped in a narrow band on both sides of the matrix diagonal, then they can be processed in packed form with only the bandwidth, or 1/2 the bandwidth stored with the necessary indexing information. Packed storage confers the following advantages:

1) Larger matrices can be stored and handled in internal storage.

2) Access time is faster with RAM; therefore, the packed form is preferred since it relieves the need for external storage.

3) A substantial amount of time is saved if the operation involving elements outside the bandwidth (zero value elements) are not performed.

4) The inverse of a given matrix expressed as a product of elementary matrices stored in packed form usually needs less storage than the explicit inverse of a matrix.

The ideal storage would be one that minimizes both the total storage used and the total computation time. In general, these two requirements are incompatible and a trade-off must be made.

In ASTRA the technique of packed storage was used to store the structure stiffness matrix [Ks] at the expense of increased running time. Instead of storing all the terms in the lower triangular matrix, only those terms within the lower half bandwidth are stored. This was necessitated by the limited amount of storage space (RAM) within the computer and the single floppy disk drive available. It was also done in order to reduce as much as possible the amount of disk storage. According to the Applesoft programming manual [6], it takes 10 bytes of computer memory to store a real number. With the floppy disk drive, a minimum of 16 bytes of storage were required to store a real number while preventing records from overwrite. In addition, to save space, some matrices made up of submatrices were selected for

storage while others were kept in RAM. An example of this is the displacement transformation matrix [R] described by Equation (3.4).

$$[R] = \begin{bmatrix} T & N & N & N \\ N & T & N & N \\ N & N & T & N \\ N & N & N & T \end{bmatrix}$$

In this matrix, [T] is a transformation matrix and [N] a null matrix. Only the non-zero terms of the primary submatrix [T] are stored on disk storage. When the matrix [R] is required by the program, [T] is recalled from storage and [R] is generated.

The features outlined above greatly extend the time required for running ASTRA, however, they allow the implementation of the program on a microcomputer. Unless the computer programs are written in machine language and optimized for the system and/or a second floppy disk drive, a hard disk drive or bubble memory becomes available as a source of storage, this is the only way in which even simple problems can be solved on the APPLE II without extensive software development and addition of RAM.

Two different types of files can be created on the APPLE II disk system; a program file and a text file. The program file is used to store the computer programs (example: ASTRA-1). The text file is more flexible and can be used to store data in sequential or random modes, and computer programs to be run from the disk.

The random storage data file are used extensively in ASTRA to handle the large matrices generated. A special feature of some of the storage subroutines in ASTRA is the storage of only non-zero real numbers. Through experimentation, it was found that it requires a minimum of 16 bytes on disk to store a real number using random access files without the risk of records overwriting one another. This applies whether the number is a large quantity or a zero. Since the floppy disk presently available can handle only 116k bytes of storage, it was imperative that the amount of data to be stored be reduced to the absolute minimum.

A feature of most matrices handled by ASTRA during the execution of the program is that they are large, sparse, symmetric matrices. By taking advantage of these characteristics, it is possible to write computer subroutines that store and retrieve the

non-zero real numbers from the lower triangular matrix only. In order to achieve this feature, advantage is taken of two features of the APPLE disk system, namely, data is stored in a random storage data file and the "on error goto" command.

The number access files on the APPLE disk are like a honeycomb, i.e., a collection of equal size cells. Each time that a record is stored in these cells, the system energizes a 250 byte block and writes the required information in it until that particular block is filled. This enables it to write information in random files, address them, and retrieve them. If in a five by five matrix the information from row three, column one is to be stored, the corresponding disk address number will be two (rows) times five (records/row) plus one (column) or disk address number eleven. For retrieval the disk will search directly the given address. When the computer tries to retrieve a record from an address that has not been used, a fatal error will be generated. When this occurs, the use of the "on error goto" command, similar to the fortran EXIT statement, will enable the program to bypass the error and continue with the program execution. The "on error goto" is used to avoid having an error message printed

and program execution halted. The command sets a flag in the computer which causes an unconditional jump to a program line indicated by a pre-established line number. Although the "on error goto" command is very helpful, extreme care must be taken when it is being used as it has side effects on the working of the computer. The use of the command causes the do loops and subroutine pointers to be disturbed. It can also cause problems with the buffers which handle the internal mathematical calculations and affect the available memory and working of the program. For these reasons it is recommended that it is used only when absolutely necessary.

The element stiffness and displacement transformation matrices are used in two different sequences of operation in ASTRA. They are first used in the calculation and assemblance of the structure of a stiffness matrix

$$[Ks] = \sum_{i=1}^{n} [R]^{T(i)} [K]^{(i)} [R]^{(i)} \qquad (3.25)$$

and in the calculation of the element forces

$$[Q]^{(i)} = [K]^{(i)} [R]^{(i)} [\bar{S}_{\alpha}] \qquad (3.33)$$

In order to speed the execution of the program, it is necessary to store both matrices $[K]^{(i)}$ and $[R]^{(i)}$ for each element. However, the storage of these matrices would take up a considerable amount of the available disk storage space. Advantage was taken of the special characteristics of each matrix for more efficient operation. For example, the element transformation matrix is made up of a series of real and null matrices

$$[R] = \begin{bmatrix} T & N & N & N \\ N & N & N & N \\ N & N & T & N \\ N & N & N & N \end{bmatrix} \quad \text{(rod elements)}$$

and

$$[R] = \begin{bmatrix} T & N & N & N \\ N & T & N & N \\ N & N & T & N \\ N & N & N & T \end{bmatrix} \quad \text{(beam elements)}$$

The submatrix [T] is a three by three real matrix while [N] is a null matrix. In order to save on storage space, only the non-zero elements of the transformation

matrix [T] are stored on the disk. Separate subroutines are then used to generate the displacement transformation matrix R according to the type of element being used.

A similar technique is used in the storage of the element stiffness matrix [K]. For a rod element, see Table III, only a few non-zero terms need to be stored. The complete matrix can be generated from a single term AE/L which is stored on the disk memory.

For beam elements, the stiffness matrix is of a more complex format and must be stored differently. A subroutine was written that stores the non-zero terms of the lower triangular matrix. This procedure takes a somewhat longer retrieval time than that used for storage of the full matrix as the full matrix must be generated from the values stored. However, this procedure has the advantage that a substantial amount of disk memory is saved when compared to storing the complete twelve by twelve matrix.

MULTIPLICATION SUBROUTINES

Multiplication of two matrices [C] = [A] [B], where [A] is an "l x m" matrix and [B] is an "m x n" matrix, is defined by

$$C_{ij} = \sum_{k=1}^{m} A_{ik} B_{kj}$$

The number of mathematical operations required to complete the multiplication is of the order of "2 x 1 x m x n". Since many of the matrices with which ASTRA deals with are made up of a combination of real and null submatrices it is convenient to write specific matrix multiplication subroutines for them. For example, in the calculation of the structure stiffness and load matrices for the different elements,

$$[K_S] = \sum_{i=1}^{n} [R]^{T(i)} \quad [K]^{(i)} \quad [R]^{(i)} \tag{3.25}$$

and

$$[Q]^{(i)} = [K]^{(i)} \quad [R]^{(i)} \quad [\overline{S}_{\alpha}] \tag{3.33}$$

special subroutines were written instead of using a general routine. The added software required is more than offset by substantial time savings in the multiplication operations when compared to a full multiplication using all terms.

Solutions of Simultaneous Equations

The method used in ASTRA for the solution of simultaneous equations is the well known Cholesky's method of triangular decomposition. This method was chosen because row or column interchanges are not required, keeping round-off errors small. In addition, this method considerably reduces the number of mathematical operations required to accomplish the matrix inversion. For most applications of finite element theory, the stiffness matrix [K] has the important characteristics of being positive definite and symmetric. With these properties, Cholesky's method is optimal as it requires only $\frac{n^3}{6} + \frac{3}{2} n^2 + \frac{7}{3} n$ operations, where n is the order of the matrix, which is approximately one fourth the number required by Gaussian Elimination.

Node Point Numbering to Exploit Sparsity

For a given structure all numbering schemes lead to the same size of stiffness matrix [K], however, different numbering schemes lead to different arrangements of non-zero terms. In order to obtain the smallest possible bandwidth it is recommended that the

structure be numbered across the "short axis," i.e., the side with the least amount of nodes, as this will produce the narrowest banded diagonal possible. For an example, see Figure 12. In ASTRA the members are also automatically redefined by the CPU with node I set equal to the number of the lower node and node J set equal to the higher node value. This is done in order to facilitate the calculation and to avoid adding a stiffness element above the diagonal. The node number is not changed, only the order of input of the nodes.

Chapter 6

RECOMMENDED IMPROVEMENTS TO ASTRA

Recommended improvements to this first version of the computer program ASTRA can be categorized into four general areas:

1. Equipment
2. Programming language
3. Computer program improvements
4. Pre and post processor subroutines

Equipment

The minimum equipment that is required to run ASTRA in any other than a laboratory environment is a 48K APPLE II or similar microcomputer, two floppy disk drives, a printer, and a TV monitor. Two floppy disk drives are a minimum because one should be used to store the programs and input information, and the other should be used to store the data generated.

At the time of this writing, the APPLE Computer Corporation has announced the introduction of the APPLE III series of microcomputers. This would be an ideal computer around which to build a new system as the amount of memory would be increased from 48K bytes in

the APPLE II to 120K bytes. The APPLE III has an integral disk operating system (DOS 3.3) with a single floppy disk that has a 20 percent (16 versus 13 sectors/ track) increased capacity over the old disk drive (DOS 3.2). In addition, it has built-in series and parrallel interface boards and allows printing of 80 characters on the monitor versus 40 on the APPLE II.

Upgrading the data storage/retrieval system is also an area of consideration for improvements to ASTRA. The addition of more low speed memory in the form of disk drives would greatly increase the capacity of the program to handle larger structures and speed the program execution by allowing the storage of more complete matrices and other generated data. This can be achieved in any of three ways: addition of more floppy disk drives (up to a maximum of 8), replacement of the floppy disk drives by a hard disk, or a bubble memory unit. The addition of more floppy disk drives is not recommended as it would cost a substantial amount of money, increase the complexity of the system, require software changes, and would not result in a worthwhile improvement in the size of memory as could be obtained by other means. Instead, the preferred alternative is the replacement of the floppy disk drive by a hard disk

or bubble memory. Although the initial cost may be greater, it is believed that in the long run it would be more economical and trouble free. Memory storage would then be increased to 10M bytes allowing the storage of the complete element stiffness and transformation matrices and other generated data. This would increase the speed of execution of the program and improve the quality of the program printout by allowing more information regarding the analysis completed to be shown.

The printer should be a commercially available matrix dot printer capable of both printing and drawing graphics so as to increase its versatility and drive down the costs and complexity associated with a separate printer and plotter unit. Graphics-capable printers are not as accurate as table plotters but are quite adequate for most applications. Table plotters are more accurate and neat and have the added option that graphics can be plotted in color. However, they are expensive to purchase and maintain and usually require the development of specialized software for their operation. A thermal type printer is not recommended unless it has graphics capability and is used as a plotter only since the quality of print is not durable

and the cost of the paper makes it prohibitive to print large quantities of computer output.

The TV monitor should be a color monitor as it can be used to advantage when implementing computer graphics. Also, a worthwhile addition to any computer system would be a digitizer. The digitizer would allow for geometric data to be entered directly from the board by means of an analog pencil and would eliminate the need for typing in the data.

Programming Language

The change in programming language from "Applesoft" Basic to UCSD Pascal is highly desirable. "Applesoft" Basic is a good lower level programming language. However, programs written in "Applesoft" are, when compared to other higher level programming languages such as Pascal or FORTRAN, bulky to store and highly inefficient in computing, output handling, and editing operations.

The substitution of "Applesoft" with Pascal would accomplish the following changes:

1. Increase amount of available RAM by 16K bytes through the use of a language card.

2. Provide for better utilization of RAM. Programs written in Pascal are said to use 50 percent less RAM to store the program than those written in "Applesoft."

3. Reduction in running time. Pascal is reputed to run up to twenty times as fast as "Applesoft."

4. Increased program editing and output format capacity.

5. Compatibility of the program with other systems having Pascal as a user's language.

Program Improvements

This first version of ASTRA is simple in scope and needs improvements in order to enhance its ability to handle more complex problems faster. In order to improve the program, it is first suggested that the computer system used be upgraded to include a minimum of two floppy disk drives or a hard disk drive, and UCSD Pascal programming language. If these changes are not implemented, it is believed that the improvements to be accomplished will not be worth the time spent in carrying them out.

These are five areas in which ASTRA can be improved:

1. The number and types of structural elements available

2. The capability to conduct thermal and dynamic analysis

3. The capacity of the program to handle more degrees of freedom

4. The program execution speed

5. Pre and post-processing

The addition of more types of structural elements to include membrane, plate, and three dimensional isoparametric elements would greatly increase the versatility of the program. In this present version, ASTRA can only handle rod and beam elements. Many structures must thus be idealized to a large extent in order to allow their analysis using ASTRA. The addition of the above mentioned elements would allow for the analysis of more varied types of structures while allowing for more accurate analysis to be accomplished.

A worthwhile research topic would be to explore the additions of options to perform thermal, dynamic and non-linear analysis. Adding the capability to ASTRA of

performing thermostructural analysis should not present much trouble as the equations are simple in nature. The addition of dynamic and non-linear analysis would be a more difficult project as it is unknown at this time whether the accuracy of the APPLE II would be sufficient to solve the necessary equations and extract the appropriate eigenvalues in other than very elementary structures.

In order to increase the capacity of the program to handle more degrees of freedom, the first steps mandated would be to convert the USCD Pascal and employ a hard disk. It is not known at this time what the absolute capacity of the APPLE II would be in terms of degrees of freedom when using Pascal. How much remaining RAM would be available after the program is compiled for the storage of matrices is the key question. When using a hard disk, the amount of time necessary to solve the problem rather than the amount of available memory may well become the deciding factor in limiting the program capacity.

Data retrieval and disk space requirements in data storage/retrieval operations can also be reduced by storing the arrays using a combination of sequential and random access files. Data arrays which do not require

access at a particular location can be stored in disk using sequential file storage. This eliminates the problem of records overwriting reduces the number of bytes required for storage by not having to use a specified record length and simplifies the software by eliminating the need for calculating the addresses of records stored in the disk.

ASTRA is sadly deficient in the availability of pre and postprocessor subroutines. These are subroutines that would help the user prepare and check the input data and then interpret the program results. These tasks are usually accomplished in the larger finite element programs by element and nodal point coordinate generating subroutines, and plotting programs that draw the structure and its deformed shape, and temperature and stress profiles. When programming plotting subroutines, the user is warned that due to the allocation of memory by the APPLE II to graphics, overwrites between the graphics data and information stored in arrays may occur, and some of the information stored in the arrays may be lost. Thus, care must be exercised when using the APPLE II high-resolution graphics capabilities.

Chapter 7

USER'S GUIDE TO ASTRA

The Subroutine IASTRA

The subroutine IASTRA (Input to ASTRA) was written for the specific purpose of helping the user prepare the input data to ASTRA. This section of the thesis is intended to serve as a user's guide to both the preprocessor IASTRA and the program ASTRA. As these are general computer codes intended for use with a variety of computer systems, no attempt will be made to tailor the user's instructions to a particular system. Instead, a comprehensive review of the preprocessor, IASTRA presented together with a general outline for running ASTRA will be described.

In order to prepare the input data to ASTRA, all that is required of the user is that he answers the questions posed by the preprocessor IASTRA. The first step in running ASTRA then is to load and run IASTRA (enter the input data). IASTRA will then require the user to select an option by entering the needed option number (and then hitting the return key), there are five user's options in the preprocessor:

1. Create a new data file

2. List input data

3. Modify an existing data file

4. Copy an existing data file

5. Exit

After the option is completed, the program will return to wait for the next option number. Option five, "EXIT," releases the program. At that time ASTRA1, ASTRA2, ASTRA3, and ASTRA4 should be loaded and run.

Option No. 1: Create a New Data File

As the name implies, this option is used to form a new set of input data. The following is a list of terms asked by the program when using the option.

1. DWG ID CODE?

   This is a code number or name under which the input data will be stored on the disk. The code can consist of a combination of alphanumeric characters less than 10 in number.

2. DATE?

   Self explanatory. Date $\leq$ 10 characters

3. YOUR NAME?

   Self explanatory. Name $\leq$ 10 characters

4. *NUMBER OF ELEMENT CARDS?*

Total number of element generating cards

5. DATA FOR ELEMENT CARDS?

The element generator cards are used to enter information about the structure elements. One card can be used to generate more than one element. The information on the element cards includes eight fields of data as shown below andin Table (IV).

Card # A B C D E F G

where

A = element type (1 = rod, 2 = beam)

B = group number

C = number of elements to be generated by the card

D = node i

E = increment for node i to be used in the generation of other elements

F = node j

G = increment for node j to be used in the generation of other elements

Example: Generate the element cards for the structure shown in Figure (16b).

Let the material and geometry of elements 2 through 5 be the same as element 1, and those of elements 7 through 13 be the same as element 6 respectively. This structure can be divided into two material groups. Group one with elements 1 through 5, and group 2 with elements 6 through 13 respectively. The element cards will look as shown in Table 4 in the Appendix.

6. ENTER MAXIMUM NUMBER OF NODES?

   Maximum number of nodes in the mesh

7. NODAL POINTS COORDINATES

   This card consists of five fields as shown below

   Card # A B C D

where

A = nodal point number

B = X coordinate

C = Y coordinate

D = Z coordinate

All coordinates are in the global or structure coordinate system.

8. NUMBER OF SUPPRESSION CARDS?

Each node number that requires to have some or all of its degrees of freedom restricted must have one suppression card. The number required at this step is the total number of suppression cards.

9. INPUT SUPPRESSIONS?

The computer will ask for the node number and what degrees of freedom are to be suppressed. Enter the node number and answer "Y" or "N" to the questions. Default is "N" (free to deflect).

10. INPUT MATERIALS AND SECTION PROPERTIES?

This part requires that the user enters the total number of material groups and then the specific material properties required for each group. In each structure there can be a number of structural elements possessing the same material and geometric characteristics. In order to avoid entering one material and geometric data card for each element, those having the same properties are given the same group number and corresponding mechanical and geometric properties entered for the group.

11. TOTAL LOAD CARDS?

For every node that has one or more forces or moments acting on it, a load card must be prepared. Loads are references on the global or structure coordinate system. A right hand system of coordinates is used with the applied load being positive when acting in the positive direction of the corresponding coordinate system.

At this point the input option is complete and the program saves the information in the proper format in the disk. For a description of the procedure used to correct errors see option NO. 3, "MODIFY OPTION."

Option No. 2: List Option

This option will print the input information of any given file on the system printer. After the option has been completed, the printer must be reset manually. In order to run this option, the printer is first energized and then the option is executed.

Option No. 3: Modify-Edit Option

As the name implies, this option is used to change an existing data file. Its uses include

correction of errors, and/or additions and deletions of nodes, element material properties, nodal suppressions, and nodal loads. This option is easy to use and all that is usually required is reentering the complete data card to be changed or added to the data file. The procedure for deleting data, for example an element card, is somewhat tricky and will be discussed below. In order to delete a data card, the program will discard the last data card for that particular section. For instance, let us imagine a set of element cards that contain five cards. If the delete option is exercised, the program will discard element No. 5. However, if the card that we want to discard is card No. 2, the procedure is somewhat different and must be executed in two steps as follow:

1. Change element card No. 2 by entering the last card in the section as card No. 2.

2. Exercise the delete option to discard the last card in the section.

The procedure described was necessitated by the limited amount of time available to develop the appropriate software. It is recommended that this procedure be simplified in the future in order to avoid confusion.

Option No. 4: Data File Duplication

This option is used to duplicate an existing data file onto another disk. The file will be saved with the existing file name. The computer will ask the user to place in the disk drive the disk containing the file to be duplicated, and the file name. Once this is done it will ask that a new disk be inserted in the disk drive and, upon hitting return, it will duplicate the file in the new floppy disk that was just inserted. Once the duplicating procedure is completed the program can be restarted by hitting return. It is recommended that all data files in use be duplicated in order to avoid loss of data if the disk containing the original file is lost or its information destroyed.

Option No. 5: Exit Option

On the completion of any of the previously discussed options, the program executes a loop and restarts itself. The EXIT option causes the program to exit the loop and thus terminates the execution of IASTRA.

BIBLIOGRAPHY

BIBLIOGRAPHY

Apple Computer Inc.
1978 Applesoft II, Basic Programming Reference Manual. Cupertino, California: Apple Computer Inc.

Apple Computer Inc.
1979 The DOS Manual. Cupertino, California: Apple Computer Inc.

Campbell, W. G. and S. V. Ballou.
1978 Form and Style Theses, Reports, Term Papers. Boston: Houghton Mifflin Company.

Cook, R. D.
1974 Concepts and Applications of Finite Element Analysis. New York: John Wiley & Sons, Inc.

Hender, D. R.
1976 Engineering Considerations in the Finite Element Heat Conduction Method. Long Beach: California State University, Long Beach.

Hsieh, Yuan-Yu
1970 Elementary Theory of Structures. Englewood Cliffs: Prentice-Hall, Inc.

Leventhal, Lance A.
1979 6502 Assembly Language Programing. Berkely: Osborne/McGraw-Hill.

Lindgren, Lars-Erik
1980 M-FEMP A Finite Element Miniprogram for CP/M Compatible Microcomputers. Lulea, Sweden: University of Lulea.

Martin, H. C. and Graham F. Carey
1973 Introduction to Finite Element Analysis. New York: McGraw-Hill Book Company.

Osborne, Adam
1980 An Introduction to Microcomputers. Berkeley: Osborne/McGraw-Hill.

Przemieniecki, J. S.
1968 Theory of Matrix Structural Analysis. New York: McGraw-Hill, Inc.

Rockey, K. C., Evans, H. R., Griffith, D. W. and D. A. Nethercot
1975 The Finite Element Method, a Basic Introduction. London: Granda Publishing Limited.

Segerlind, Larry J.
1976 Applied Finite Element Analysis. New York: John Wiley & Sons, Inc.

Tewarson, R. P.
1974 Sparce Matrices. New York: Academic Press, Inc.

Timoshenko, S. P. and J. N. Goodier
1970 Theory of Elasticity. New York: McGraw-Hill Book Co., Inc.

Volterra, E. and J. H. Gaines
1971 Advanced Strength of Materials. Englewood Cliffs: Prentice Hall Inc.

Weaver, W.
1967 Computer Programs for Structural Analysis. *Princeton: D. Van Nostrand Company, Inc.*

Zienkiewicz, O. C., and Y. K. Cheung
1967 The Finite Element Method in Structural and Continuum Mechanics. London: McGraw-Hill Publishing Company Limited.

APPENDICES

APPENDIX A

Table 1

Stiffness Matrix K for a Beam Element with Shear Deformations

$$
\begin{bmatrix}
\frac{EA}{l} & & & & & & & & & & & \\
0 & \frac{12EI_z}{l^3(1+\Phi_y)} & & & & & & & & & & \\
0 & 0 & \frac{12EI_y}{l^3(1+\Phi_z)} & & & & & & & & & \\
0 & 0 & 0 & \frac{GJ}{l} & & & & & & & \text{Symmetric} & \\
0 & 0 & \frac{-6EI_y}{l^2(1+\Phi_z)} & 0 & \frac{(4+\Phi_z)EI_y}{l(1+\Phi_z)} & & & & & & & \\
0 & \frac{6EI_z}{l^2(1+\Phi_y)} & 0 & 0 & 0 & \frac{(4+\Phi_y)EI_z}{l(1+\Phi_y)} & & & & & & \\
\frac{-EA}{l} & 0 & 0 & 0 & 0 & 0 & \frac{AE}{l} & & & & & \\
0 & \frac{-12EI_z}{l^3(1+\Phi_y)} & 0 & 0 & 0 & \frac{-6EI_z}{l^2(1+\Phi_y)} & 0 & \frac{12EI_z}{l^3(1+\Phi_y)} & & & & \\
0 & 0 & \frac{-12EI_y}{l^3(1+\Phi_z)} & 0 & \frac{6EI_y}{l^2(1+\Phi_z)} & 0 & 0 & 0 & \frac{12EI_y}{l^3(1+\Phi_z)} & & & \\
0 & 0 & 0 & \frac{-GJ}{l} & 0 & 0 & 0 & 0 & 0 & \frac{GJ}{l} & & \\
0 & 0 & \frac{-6EI_y}{l^2(1+\Phi_z)} & 0 & \frac{(2-\Phi_z)EI_y}{l(1+\Phi_z)} & 0 & 0 & 0 & \frac{6EI_y}{l^2(1+\Phi_z)} & 0 & \frac{(4+\Phi_z)EI_y}{l(1+\Phi_z)} & \\
0 & \frac{6EI_z}{l^2(1+\Phi_y)} & 0 & 0 & 0 & \frac{(2-\Phi_y)EI_z}{l(1+\Phi_y)} & 0 & \frac{-6EI_z}{l^2(1+\Phi_y)} & 0 & 0 & 0 & \frac{(4+\Phi_y)EI_z}{l(1+\Phi_y)}
\end{bmatrix}
$$

Table 2

Stiffness Matrix K for a Beam Element without Shear Deformations

$$
\begin{array}{cccccc|cccccc}
\frac{EA_X}{L} & 0 & 0 & 0 & 0 & 0 & -\frac{EA_X}{L} & 0 & 0 & 0 & 0 & 0 \\
0 & \frac{12EI_Z}{L^3} & 0 & 0 & 0 & \frac{6EI_Z}{L^2} & 0 & -\frac{12EI_Z}{L^3} & 0 & 0 & 0 & \frac{6EI_Z}{L^2} \\
0 & 0 & \frac{12EI_Y}{L^3} & 0 & -\frac{6EI_Y}{L^2} & 0 & 0 & 0 & -\frac{12EI_Y}{L^3} & 0 & -\frac{6EI_Y}{L^2} & 0 \\
0 & 0 & 0 & \frac{GI_X}{L} & 0 & 0 & 0 & 0 & 0 & -\frac{GI_X}{L} & 0 & 0 \\
0 & 0 & -\frac{6EI_Y}{L^2} & 0 & \frac{4EI_Y}{L} & 0 & 0 & 0 & \frac{6EI_Y}{L^2} & 0 & \frac{2EI_Y}{L} & 0 \\
0 & \frac{6EI_Z}{L^2} & 0 & 0 & 0 & \frac{4EI_Z}{L} & 0 & -\frac{6EI_Z}{L^2} & 0 & 0 & 0 & \frac{2EI_Z}{L} \\
\hline
-\frac{EA_X}{L} & 0 & 0 & 0 & 0 & 0 & \frac{EA_X}{L} & 0 & 0 & 0 & 0 & 0 \\
0 & -\frac{12EI_Z}{L^3} & 0 & 0 & 0 & -\frac{6EI_Z}{L^2} & 0 & \frac{12EI_Z}{L^3} & 0 & 0 & 0 & -\frac{6EI_Z}{L^2} \\
0 & 0 & -\frac{12EI_Y}{L^3} & 0 & \frac{6EI_Y}{L^2} & 0 & 0 & 0 & \frac{12EI_Y}{L^3} & 0 & \frac{6EI_Y}{L^2} & 0 \\
0 & 0 & 0 & -\frac{GI_X}{L} & 0 & 0 & 0 & 0 & 0 & \frac{GI_X}{L} & 0 & 0 \\
0 & 0 & -\frac{6EI_Y}{L^2} & 0 & \frac{2EI_Y}{L} & 0 & 0 & 0 & \frac{6EI_Y}{L^2} & 0 & \frac{4EI_Y}{L} & 0 \\
0 & \frac{6EI_Z}{L^2} & 0 & 0 & 0 & \frac{2EI_Z}{L} & 0 & -\frac{6EI_Z}{L^2} & 0 & 0 & 0 & \frac{4EI_Z}{L}
\end{array}
$$

Table 3

Stiffness Matrix for a Space Rod Element

$$\left[\frac{AE}{L}\right]\begin{bmatrix} 1 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Table 4

Sample Element Generating Cards

CARD #	ELEMENT TYPE	GROUP #	# OF ELMTS	NODE i	INCREMENT IN i	NODE J	INCREMENT IN J
1	2	1	2	4	1	5	1
2	2	1	2	7	1	8	1
3	2	1	1	10	0	11	0
4	2	2	3	1	3	4	3
5	2	2	3	2	3	5	3
6	2	2	3	3	3	6	3

Table 5

A Comparison of Features of Some Microcomputers

COMPANY	MODEL	RAM	ROM	TERMINAL I/O	STORAGE	SOFTWARE	COMMENTS
APPLE COMPUTER	APPLE II	64K	8K	COLOR RF MONITOR 40X24 CHAR KEYBOARD	5 INCH FLOPPY, CASSETTE 10 MB DISK	MONITOR BASIC PASCAL COBOL APL FORTRAN	LOW AND HIGH RESOLUTION GRAPHICS GOOD DOCUMENTATION AVAILABLE, VERY POPULAR MICROCOMPUTER, WIDE RANGE OF SOFTWARE
COMMODORE	PET 2001	32K	14K	9" DIPLAY 40X25 CHAR 73 KEYS	CASSETTE 5 INCH FLOPPY	MONITOR BASIC	NON-STANDARD SIZE KEYBOARD, POOR DOCUMENTATION
CROMENCO	SYSTEM 2	32K	--		5 INCH FLOPPY IBM COMP	MONITOR BASIC FORTRAN	TURNKEY MAINFRAME SYS USES S100 BUS
EXIDY	SORCERER	32K	4K	RF VIDEO 64X30 CHAR KEYBOARD	CASSETTE	MONITOR BASIC	HIGH RESOLUTION GRAPHICS
IMSAI	VDP40	64K	2K	9" DISPLAY 80X24 CHAR 62 KEYS	5 INCH FLOPPY	MONITOR ASSEMBLER BASIC	USES S100 BUS
NORTH STAR	HORIZON	48K	--	VIDEO TERMINAL	5 INCH FLOPPY	MONITOR FORTRAN PASCAL	MAINFRAME SYSTEM USES S100 BUS
OHIO SCIENT.	C3-B	48K	--	VIDEO TERMINAL	5 INCH FLOPPY, HARD DISK	BASIC FORTRAN COBOL ASSEMBLY	HAS THREE TYPES OF PROCESSORS; Z80, 6800 AND 6502 HARD DISK AVAILABLE
RADIO SHACK	TRS-80	32K	12K	VIDEO 64X16 CHAR KEYBOARD	CASSETTE 5 INCH FLOPPY	MONITOR BASIC FORTRAN	Z80 BOARD, DOCUMENTATION AVAILABLE WIDE RANGE OF SOFTWARE

APPENDIX B

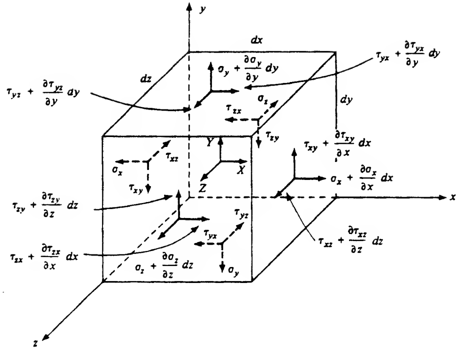


Figure 1

Body and Surface Forces Acting
on a Small Cube

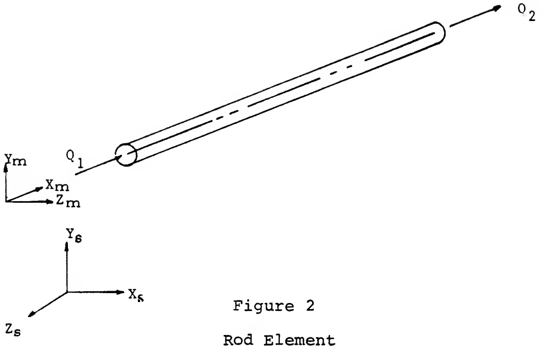


Figure 2

Rod Element

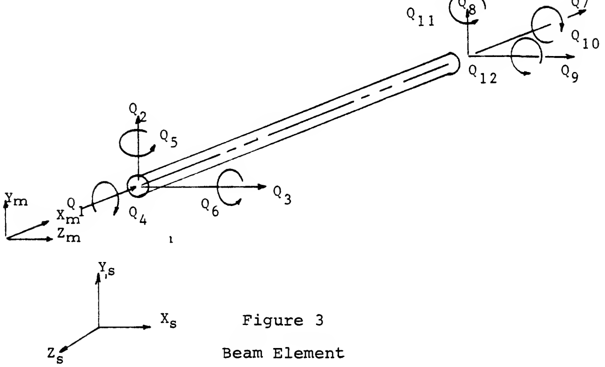


Figure 3

Beam Element

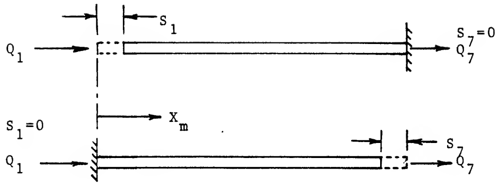


Figure 4

Axial Displacement

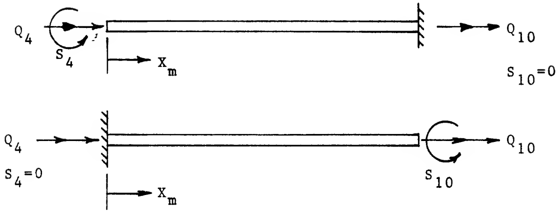


Figure 5

Twisting Displacement

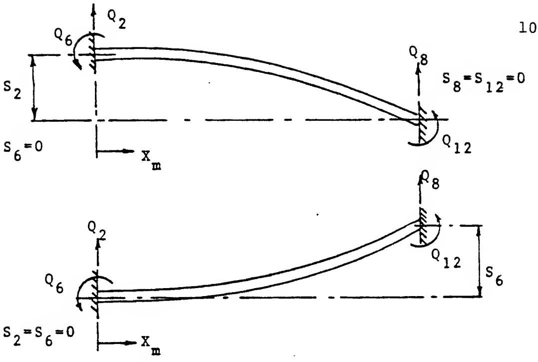


Figure 6

Displacement Due to Shear

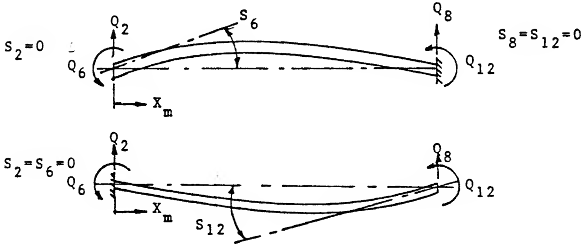


Figure 7

Displacements Due to Bending Moments

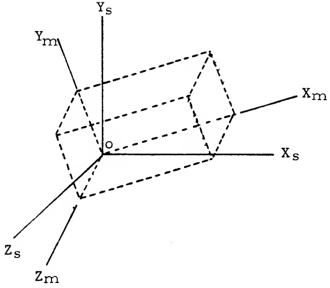


Figure 8

Coordinate Systems

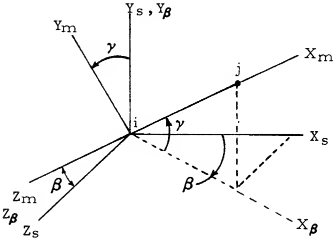


Figure 9

Rotation of Axes for a Space Rod

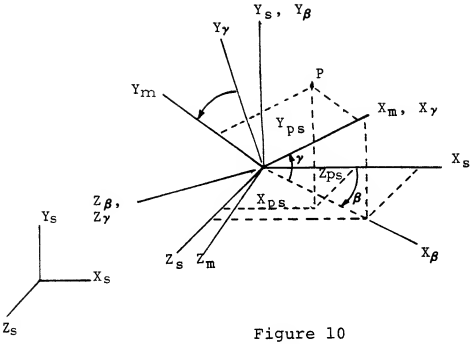


Figure 10

Rotation of Axes for a Space Beam

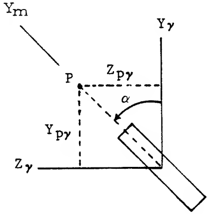
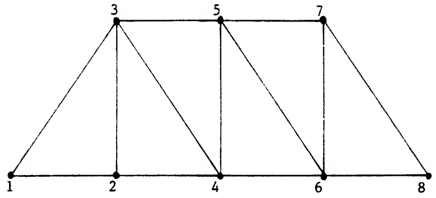
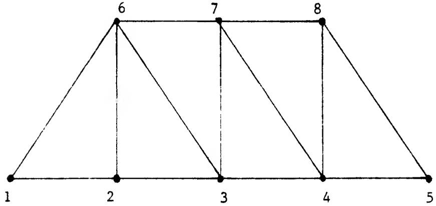


Figure 11

Rotation of a Space Beam About X_m Axis



(A) desirable



(B) less desirable

Figure 12

Node Point Numbering

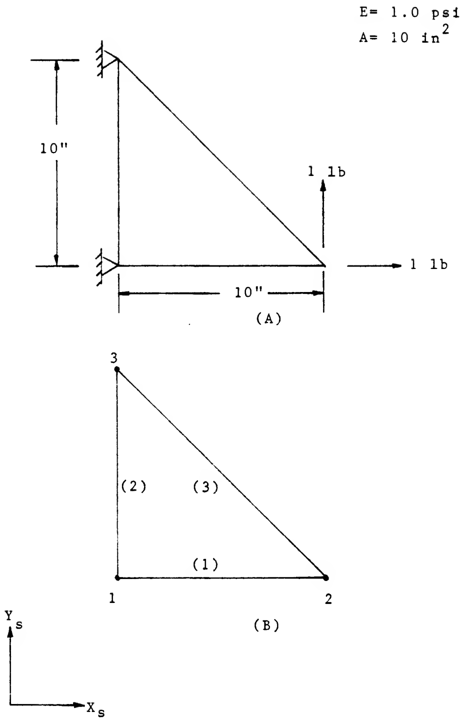


Figure 13

Example Problem No. 1, Plane Truss

E= 10ksi
A_1= 6 sq in
A_2= 8 sq in
A_3= 10 sq in

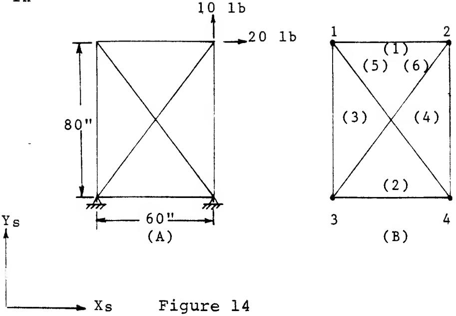


Figure 14

Example Problem No. 2, Plane Truss

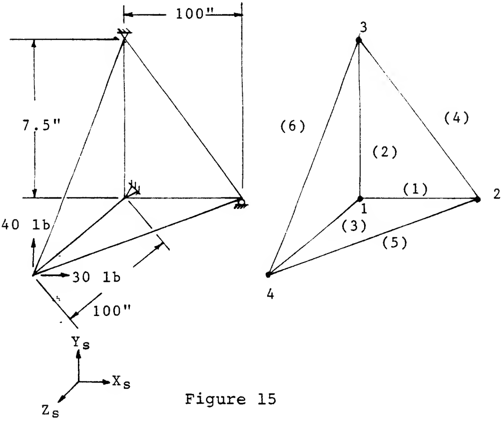


Figure 15

Example Problem No. 3, Space Truss

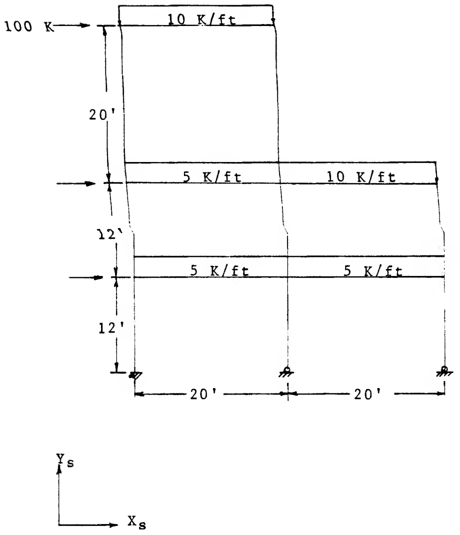


Figure 16

Example Problem No. 4, Plane Frame

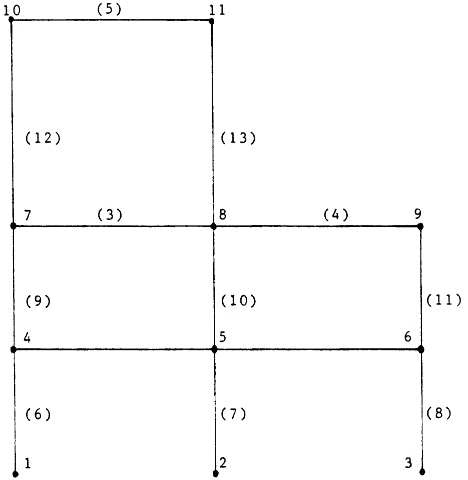


Figure 17

Example Problem No. 4, Frame Modeled

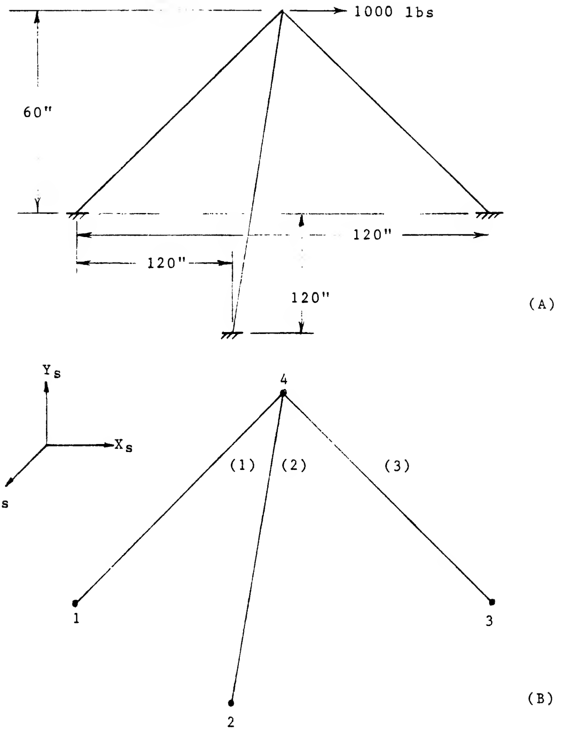
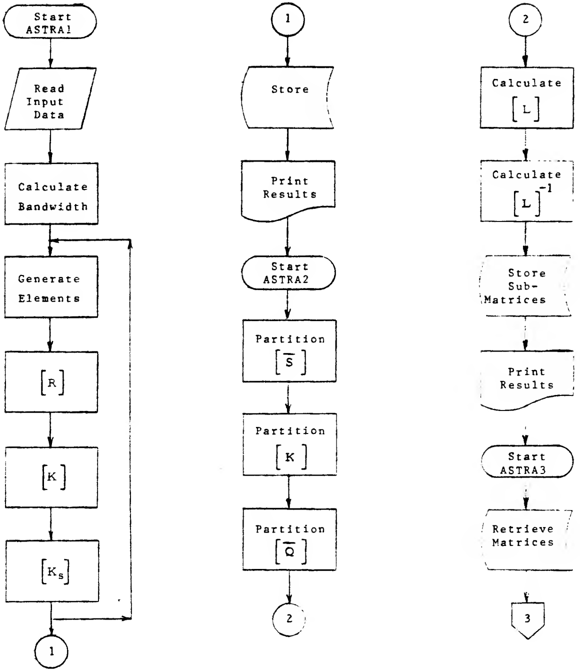


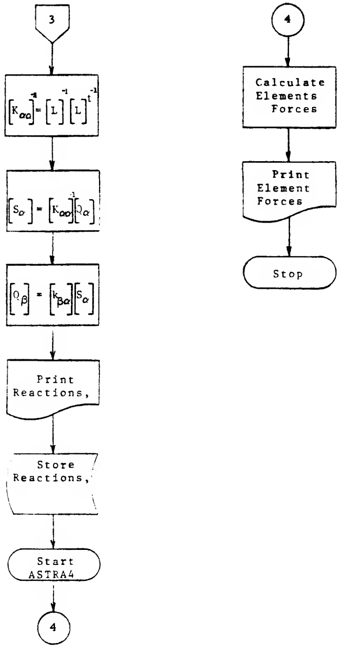
Figure 18

Example Problem No. 5, Space Frame

APPENDIX C

Program Flowchart for ASTRA




3
$[K_{\alpha\alpha}]^{-1} = [L]^{-1} [L]^{t^{-1}}$
$[S_\alpha] = [K_{\alpha\alpha}]^{-1} [Q_\alpha]$
$[Q_\beta] = [k_{\beta\alpha}] [S_\alpha]$
Print Reactions,
Store Reactions,
Start ASTRA4
4
4
Calculate Elements Forces
Print Element Forces
Stop

APPENDIX D

```
]LIST

10  REM  6 DEC 1980
20  REM  PROGRAM IASTRA
30  HOME : CLEAR
40  GOSUB 300: REM  ALLOCATE ARRAYS
50  HTAB (10): PRINT "PROGRAM IASTRA"
60  PRINT
70  HTAB (5): PRINT "ANALISING STRUCTURES WITH APPLE"
80  PRINT
90  VTAB (5): PRINT "THIS PROGRAM IS USED TO PREPARE THE INPUT DATA TO THE
     FINITE ELEMENTS ANALYSIS COMPUTER PROGRAM ASTRA"
100  VTAB (9): PRINT "CHOOSE ONE OF THE AVAILABLE OPTIONS:"
110  VTAB (11): PRINT  TAB( 5);"1. CREATE A NEW INPUT DATA FILE."
120  PRINT  TAB( 5);"2. LIST INPUT DATA.
130  PRINT  TAB( 5);"3. MODIFY AN EXISTING DATA FILE."
140  PRINT  TAB( 5);"4. COPY OPTION."
160  PRINT  TAB( 5);"5. EXIT FROM THE PROGRAM."
170  VTAB (19): INPUT "ENTER OPTION NUMBER ?";A%
180  IF (A% < 1) OR (A% > 6) THEN  HOME : PRINT "WRONG OPTION NUMBER, TRY
     AGAIN": GOTO 100
190  HOME : IF A% = 1 THEN  GOSUB 380: REM  INPUT OPTION
200  IF A% = 2 THEN  GOSUB 3870: REM  LIST OPTION
210  IF A% = 3 THEN  GOSUB 5030: REM  MODIFY OPTION
220  IF A% = 4 THEN  GOSUB 4920: REM  COPY OPTION
240  IF A% = 5 GOTO 270: REM  EXIT
250  HOME : CLEAR
260  GOTO 10
270  HOME : PRINT "END OF JOB"
280  END
290  REM   ***************************************************************
     ******************************************************************
     ******************
300  REM  ALLOCATION OF ARRAYS
310  DIM AAID$(5),ABEGEN%(100,9)
320  DIM ACNPCO(200,3)
330  DIM AGNDOF%(50,6): REM  NODAL SUPPRESSIONS
340  DIM AJPNODE(200,6): REM  LOAD AT NODES
350  DIM AKMSPROP(10,9): REM  MAT & SECTION PROPERTIES.
360  RETURN
370  REM   ***********************************************
380  REM  SUBROUTINE INPUT
390  HOME : VTAB (3)
400  GOSUB 3000: REM   INPUT HEADING
420  HOME : PRINT "ENTER THE FOLLOWING INFORMATION FOR EACH ELEMENT GENERA
     TOR CARD IN FREE FORMAT."
430  VTAB (7): PRINT "NOTE:  ANY ERRORS NOT CORRECTED BEFORE  THE LINES AR
     E ENTERED CAN BE TAKEN CARE OF AFTER ALL CARDS HAVE BEEN ENTERED.": PRINT

440  PRINT "HIT RETURN TO CONTINUE"
445  INPUT A$
460  HOME
470  GOSUB 3380: REM    ELEMENT CARDS
480  GOSUB 3680: REM   NPCOS
490  GOSUB 560: REM  INPUT SUPPRESSIONS
500  GOSUB 1010: REM   INPUT MAT. & SECT. PROP.
510  GOSUB 1390: REM  INPUT LOADS
520  HOME
530  PRINT "INPUT SUBROUTINE COMPLETED"
540  GOSUB 1750: REM   SAVE DATA
```

```
550  RETURN
560  REM   ******************************************
570  REM  INPUT SUPPRESSIONS
580  HOME
590  INPUT "NUMBER OF SUPPRESSIONS CARDS ? ";AH%
600  IF AH% > 100 THEN  PRINT "TOO MANY NODES SUPPRESSED (N<=100); TRY AGA
     IN": PRINT : GOTO 590
610  HOME
620  VTAB (5): PRINT "ENTER OPTION NUMBER"
630  VTAB (7): PRINT  TAB( 3);"1.  ONLY ROD ELEMENTS ARE USED."
640  PRINT  TAB( 3);"2.  A MIXTURE OF BEAM AND ROD ELEMENTS ARE USED."
650  INPUT "ENTER OPTION NUMBER ? ";I1%
660  IF (I1% < 1) OR (I1% > 2) THEN  HOME : PRINT "WRONG OPTION NUMBER; TR
     Y AGAIN ": VTAB (3): GOTO 620
670 I2% = 1
680  GOSUB 790
690  HOME
700  INPUT "DO YOU WANT TO ADD ANY MORE SUPPRESSIONS ?(Y OR N) ";A$
710  IF (A$ = "N") GOTO 760
720  IF (A$ <  > "Y") THEN  PRINT "WRONG INPUT": GOTO 700
730  INPUT "HOW MANY EXTRA NODES?";I3%
740 I2% = AH% + 1:AH% = AH% + I3%
750  GOTO 680
760  RETURN
770  REM   ************************************
780  REM  INPUT SUPPRESSIONS-1
790  FOR I = I2% TO AH%
800  HOME : PRINT "ENTER NODE NUMBER AND Y OR N TO THE QUESTIONS.": VTAB (
     3)
810  PRINT
820  PRINT "NODAL SUPPRESSIONS CARD # ... ";I
830  VTAB (6)
840  INPUT "NODE NUMBER ? ";AG%(I,0)
850  INPUT "SUPPRESS X ? ";I$
860  IF (I$ = "Y") THEN AG%(I,1) = 1
870  INPUT "SUPPRESS Y ? ";I$
880  IF (I$ = "Y") THEN AG%(I,2) = 1
890  INPUT "SUPPRESS Z ? ";I$
900  IF (I$ = "Y") THEN AG%(I,3) = 1
910  IF I1% = 1 GOTO 980
920  INPUT "SUPPRESS THETA X ? ";I$
930  IF (I$ = "Y") THEN AG%(I,4) = 1
940  INPUT "SUPPRESS THETA Y ? ";I$
950  IF (I$ = "Y") THEN AG%(I,5) = 1
960  INPUT "SUPPRESS THETA Z ? ";I$
970  IF (I$ = "Y") THEN AG%(I,6) = 1
980  NEXT
990  RETURN
1000  REM   ***************************************************
1010  HOME : PRINT "INPUT MAT. & SECT. PROPERTIES"
1020  VTAB (3)
1030  INPUT "NUMBER OF GROUPS ? ";AF%
1040  HOME
1050  VTAB (3): PRINT "ENTER OPTION NUMBER"
1060  VTAB (7): PRINT  TAB( 3);"1. ONLY ROD ELEMENTS ARE USED."
1070  PRINT  TAB( 3);"2.  A MIXTURE OF RODS AND BEAM ELEMENTS ARE USED. "
1080  VTAB (11)
1090  INPUT "ENTER OPTION NUMBER ?";I1%
1100  IF (I1% < 1) OR (I1% > 2) THEN  HOME : PRINT "WRONG OPTION NUMBER; T
      RY AGAIN ": VTAB (3): GOTO 1050
```

```
1110 I2% = 1
1120  FOR I = I2% TO AF%
1130  HOME
1140  PRINT "MAT. & SECT. PROPERTIES FOR GROUP # ";I
1150  IF I1% = 1 THEN AK(I,0) = 1: GOTO 1170
1160  INPUT "ELEMENT TYPE ? ";AK(I,0)
1170  VTAB (3): INPUT "CROSS-SECTIONAL AREA ? ";AK(I,1)
1180  IF I1% = 1 GOTO 1220
1190  PRINT "MOMENTS OF INERTIA ABOUT LOCAL AXIS": PRINT
1200  INPUT "I (Y'-Y') ? ";AK(I,2)
1210  INPUT "I (Z'-Z') ? ";AK(I,3)
1220  PRINT : INPUT "MODULUS OF ELASTICITY E ? ";AK(I,4)
1230  IF I1% = 1 GOTO 1290
1240  INPUT "POISSON'S RATIO V ? ";AK(I,5)
1250  INPUT "POLAR MOMENT OF INERTIA J ? ";AK(I,6)
1260  INPUT "X'  ? ";AK(I,7)
1270  INPUT "Y'  ? ";AK(I,8)
1280  INPUT "Z'  ? ";AK(I,9)
1290  NEXT
1300  HOME
1310  INPUT "DO YOU WANT TO ADD ANY MORE GROUPS ?(Y OR N) ";A$
1320  IF (A$ = "N") GOTO 1370
1330  IF (A$ < > "Y") THEN  PRINT "WRONG INPUT": GOTO 1310
1340  INPUT "HOW MANY EXTRA GROUPS ? ";I3%
1350 I2% = AF% + 1:AF% = AF% + I3%
1360  GOTO 1120
1370  RETURN
1380  REM   ************************************
1390  HOME : PRINT "LOADS OPTION"
1400  VTAB (3): PRINT "THE FOLLOWING INPUT PERTAINS TO THE LOAD CARDS."
1410  VTAB (6): INPUT "ENTER NUMBER OF LOAD CARDS ? ";AI%
1420  HOME
1430  VTAB (3): PRINT "LOAD OPTIONS. "
1440  VTAB (7): PRINT  TAB( 3);"1. ONLY ROD ELEMENTS ARE USED."
1450  PRINT  TAB( 3);"2.  A MIXTURE OF RODS AND BEAM ELEMENTSS ARE USED.

1460  VTAB (11): INPUT "ENTER LOAD OPTION ? ";I1%
1470  IF (I1% < 1) OR (I1% > 2) THEN  HOME : PRINT "WRONG OPTION NUMBER; T
     RY AGAIN ": GOTO 1430
1480 I2% = 1
1490  GOSUB 1590
1500  HOME
1510  INPUT "DO YOU WANT TO ADD ANY MORE LOAD CARDS ?(Y OR N) ";A$
1520  IF (A$ = "N") GOTO 1570
1530  IF (A$ <  > "Y") THEN  PRINT "WRONG INPUT": GOTO 1510
1540  INPUT "HOW MANY EXTRA CARDS?";I3%
1550 I2% = AI% + 1:AI% = AI% + I3%
1560  GOTO 1490
1570  RETURN
1580  REM   ***********************************************
1590  REM  INPUT LOAD-1
1600  FOR I = I2% TO AI%
1610  HOME
1620  PRINT "LOAD FOR LOAD CARD # ";I
1630  VTAB (6)
1640  INPUT "ENTER NODE NUMBER ? ";AJ(I,0)
1650  INPUT "ENTER P-X ? ";AJ(I,1)
1660  INPUT "ENTER P-Y ? ";AJ(I,2)
1670  INPUT "ENTER P-Z ? ";AJ(I,3)
1680  IF (I1% = 1) THEN  HOME : GOTO 1720
```

```
1690  INPUT "ENTER M-X ? ";AJ(I,4)
1700  INPUT "ENTER M-Y ? ";AJ(I,5)
1710  INPUT "ENTER M-Z ? ";AJ(I,6)
1720  NEXT
1730  RETURN
1740  REM   ***********************************************
1750  REM  SUBROUTINE SAVE DATA
1760 D$ =  CHR$ (4): REM  CTRL D
1770  PRINT D$;"OPEN ";AA$(0)
1780  PRINT D$;"DELETE";AA$(0)
1790  PRINT D$;"OPEN ";AA$(0);", L12"
1800  PRINT D$;"WRITE ";AA$(0);", R";1
1810  PRINT AD%
1820  PRINT D$;"WRITE ";AA$(0);",R ";2
1830  PRINT AE%
1840  PRINT D$;"WRITE ";AA$(0);",R";3
1850  PRINT AF%
1860  PRINT D$;"WRITE ";AA$(0);",R";4
1870  PRINT AH%
1880  PRINT D$;"WRITE ";AA$(0);",R";5
1890  PRINT AI%
1900  REM  SAVE ELEMENT CARDS
1910  FOR I = 1 TO AE%
1920 J = ((I - 1) * 9) + 10
1930  FOR K = 1 TO 9
1940  PRINT D$;"WRITE";AA$(0);",R";(J + K)
1950  PRINT AB%(I,K)
1960  NEXT K
1970  NEXT I
1980  REM  SAVE NPCO
1990 A = (AE% * 9) + 10
2000  FOR I = 1 TO AD%
2010 J = ((I - 1) * 3) + A
2020  FOR K = 1 TO 3
2030  PRINT D$;"WRITE ";AA$(0);",R";(J + K)
2040  PRINT AC(I,K)
2050  NEXT K
2060  NEXT I
2070  REM  SAVE SUPPRESSIONS
2080 A = (AE% * 9) + (AD% * 3) + 10
2090  FOR I = 1 TO AH%
2100 J = A + ((I - 1) * 7)
2110  FOR K = 0 TO 6
2120  PRINT D$;"WRITE";AA$(0);",R";(J + 1 + K)
2130  PRINT AG%(I,K)
2140  NEXT K
2150  NEXT I
2160  REM  SAVE MATERIAL & SECTION PROPERTIES.
2170 A = A + (AH% * 7)
2180  FOR I = 1 TO AF%
2190 J = A + ((I - 1) * 10)
2200  FOR K = 0 TO 9
2210  PRINT D$;"WRITE";AA$(0);",R";(J + 1 + K)
2220  PRINT AK(I,K)
2230  NEXT K
2240  NEXT I
2250  REM  SAVE LOADS
2260 A = A + (AF% * 10)
2270  FOR I = 1 TO AI%
2280 J = A + ((I - 1) * 7)
```

```
2290  FOR K = 0 TO 6
2300  PRINT D$;"WRITE";AA$(0);",R";(J + 1 + K)
2310  PRINT AJ(I,K)
2320  NEXT K
2330  NEXT I
2340  PRINT D$;"CLOSE";AA$(0)
2350  PRINT "STORAGE OF INPUT DATA COMPLETE"
2360  RETURN
2370  REM  **************************************************
2380  REM  SUBROUTINE READ DATA
2390 D$ =  CHR$ (4): REM  CTRL D
2400  GOSUB 3000
2410  HOME
2420  PRINT D$;"OPEN ";AA$(0);", L12"
2430  PRINT D$;"READ ";AA$(0);",R";1
2440  INPUT AD%
2450  PRINT D$;"READ ";AA$(0);",R";2
2460  INPUT AE%
2470  PRINT D$;"READ ";AA$(0);",R";3
2480  INPUT AF%
2490  PRINT D$;"READ ";AA$(0);",R";4
2500  INPUT AH%
2510  PRINT D$;"READ ";AA$(0);",R";5
2520  INPUT AI%
2530  REM  READ ELEMENT CARDS
2540  FOR I = 1 TO AE%
2550 J = ((I - 1) * 9) + 10
2560  FOR K = 1 TO 9
2570  PRINT D$;"READ ";AA$(0);",R";(J + K)
2580  INPUT AB%(I,K)
2590  NEXT K
2600  NEXT I
2610  REM  READ NPCC
2620 A = (AE% * 9) + 10
2630  FOR I = 1 TO AD%
2640 J = ((I - 1) * 3) + A
2650  FOR K = 1 TO 3
2660  PRINT D$;"READ ";AA$(0);",R";(J + K)
2670  INPUT AC(I,K)
2680  NEXT K
2690  NEXT I
2700  REM   READ SUPPRESSIONS
2710 A = (AE% * 9) + (AD% * 3) + 10
2720  FOR I = 1 TO AH%
2730 J = A + ((I - 1) * 7)
2740  FOR K = 0 TO 6
2750  PRINT D$;"READ ";AA$(0);",R";(J + 1 + K)
2760  INPUT AG%(I,K)
2770  NEXT K
2780  NEXT I
2790  REM  READ MAT. & SECT. PROP.
2800 A = A + (AH% * 7)
2810  FOR I = 1 TO AF%
2820 J = A + ((I - 1) * 10)
2830  FOR K = 0 TO 9
2840  PRINT D$;"READ ";AA$(0);",R";(J + 1 + K)
2850  INPUT AK(I,K)
2860  NEXT K
2870  NEXT I
2880  REM  READ LOADS
```

```
2890 A = A + (AF% * 10)
2900  FOR I = 1 TO AI%
2910 J = A + ((I - 1) * 7)
2920  FOR K = 0 TO 6
2930  PRINT D$;"READ ";AA$(0);",R";(J + 1 + K)
2940  INPUT AJ(I,K)
2950  NEXT K
2960  NEXT I
2970  PRINT D$;"CLOSE";AA$(0)
2980  RETURN
2990  REM  ********************************
3000  REM  INPUT HEADING
3010  HOME : VTAB (3)
3020  PRINT "SUBROUTINE HEADING"
3030  VTAB (10)
3040  INPUT "DWG ID CODE ?";AA$(0)
3050  INPUT "DATE ?";AAID$(1)
3060  INPUT "YOUR NAME ?";AAID$(2)
3100  RETURN
3370  REM   **********************************************
3380  REM  INPUT ELEMENT CARDS
3390  INPUT "NUMBER OF ELEMENT GENERATING CARDS ?";AECARDNUMBER%
3400 I2% = 1
3410  FOR I1 = I2% TO AE%
3420  HOME
3430  PRINT "ENTER DATA FOR ELMT CARD #  ";I1
3440  PRINT
3450  VTAB (5): INPUT "1.  ELEMENT TYPE (OPTION #) ? ";AB%(I1,1)
3460  INPUT "2.  GROUP NUMBER.";AB%(I1,2)
3470  INPUT "3.  NUMBER OF ELEMENTS ?";AB%(I1,3)
3480  INPUT "4.  NODE #1.";AB%(I1,4)
3490  IF AB%(I1,3) = 1 GOTO 3510
3500  INPUT "5.  INCREMENT #1 ?";AB%(I1,5)
3510  INPUT "6.  NODE #2. ?";AB%(I1,6)
3520  IF AB%(I1,3) = 1 GOTO 3540
3530  INPUT "7.  INCREMENT #2 ?";AB%(I1,7)
3540  IF AB%(I1,1) <  > 3 GOTO 3580: REM   #1
3550  INPUT "8.  NODE #3 ?";AB%(I1,8)
3560  IF AB%(I1,3) = 1 GOTO 3580
3570  INPUT "9.  INCREMENT #3 ?";AB%(I1,9)
3580  NEXT
3590  HOME
3600  INPUT "DO YOU WANT TO ADD MORE ELEMENT CARDS (Y OR N) ?  ";A$
3610  IF (A$ = "N") GOTO 3670
3620  IF (A$ <  > "Y") THEN  PRINT "WRONG INPUT": GOTO 3600
3630  HOME : INPUT "HOW MANY ? ";I3%
3640 I2% = AE% + 1
3650 AE% = AE% + I3%
3660  GOTO 3410
3670  RETURN
3680  REM   ********************************************
3690  HOME : REM  INPUT OF NPCO
3700  INPUT "ENTER MAX. NUMBER OF NODES (N<=50) ? ";AD%
3710 I2% = 1
3720  FOR I1 = I2% TO AD%
3730  HOME : PRINT "ENTER THE NPCO FOR NODE #  ";I1
3740  VTAB (5): INPUT "X COORDINATE ?";AC(I1,1)
3750  INPUT "Y COORDINATE ?";AC(I1,2)
3760  INPUT "Z COORDINATE ?";AC(I1,3)
3770  NEXT
```

```
3780  HOME
3790  INPUT "DO YOU WANT TO ADD ANY MORE NODES (Y OR N) ? ";A$
3800  IF (A$ = "N") GOTO 3850
3810  IF (A$ < > "Y") THEN  PRINT "WRONG INPUT": GOTO 3790
3820  HOME : INPUT "HOW MANY ? ";I3%
3830 I2% = AD% + 1:AD% = AD% + I3%
3840  GOTO 3720
3850  RETURN
3860  REM  ***************************************************************
3870  HOME : PRINT "LIST OPTION"
3880  VTAB (5): PRINT "TURN PRINTER ON"
3890  GOSUB 2380: REM   READ DATA
3900  REM_ PRINTER CONTROL
3910 P5$ = "": REM  LINE LENGTH
3920 P6$ = "": REM  NORMAL MODE
3930 P9$ = "": REM  10 CHAR/LINE
3940 PA$ = "": REM  12 CHAR/LINE
3950 PB$ = "": REM  SELECT PRINTER
3960  PR# 1
3970  PRINT P5$
3980  PRINT P6$
3990  PRINT PA$
4000  PRINT PB$
4010  PRINT "HEADING INFORMATION"
4020  PRINT : PRINT
4030  PRINT "DWG ID CODE IS .... ";AA$(0)
4050  PRINT "DATE IS ...........";AA$(1)
4060  PRINT "PREPARED BY .......";AA$(2)
4080  REM   PRINT ELEMENTS
4090  PRINT : PRINT
4100  PRINT "ELEMENT CARDS": PRINT
4110  PRINT "  LINE";
4120  HTAB (8): PRINT "ELMT  GROUP  #  OF";
4130  HTAB (27): PRINT "NODE  INC  NODE  INC  NODE  INC"
4140  PRINT "   #";
4150  PRINT  TAB( 4);"TYPE"; TAB( 4);"#"; TAB( 6);"ELMT"; TAB( 5);
4160  PRINT "1"; TAB( 5);"1"; TAB( 6);"2"; TAB( 6);"2";
4170  PRINT  TAB( 4);"3"; TAB( 4);"3": PRINT
4180  FOR I = 1 TO AE%
4190  HTAB (4): PRINT I;
4200  HTAB (9): PRINT AB%(I,1);
4210  HTAB (16): PRINT AB%(I,2);
4220  HTAB (22): PRINT AB%(I,3);
4230  HTAB (28): PRINT AB%(I,4);
4240  HTAB (33): PRINT AB%(I,5);
4250  HTAB (39): PRINT AB%(I,6);
4260  HTAB (1): PRINT  TAB( 6);AB%(I,7);
4270  HTAB (1): PRINT  TAB( 6);AB%(I,8);
4280  HTAB (1): PRINT  TAB( 6);AB%(I,9)
4290  NEXT
4300  REM  PRINT NPCO
4310  PRINT : PRINT
4320  PRINT "NODE #"; TAB( 3);"X-COORD"; TAB( 4);"Y-COORD"; TAB( 4);"Z-COO
     RD"
4330  PRINT
4340  FOR I = 1 TO AD%
4345  PRINT I;
4350  HTAB (11): PRINT AC(I,1);
4360  HTAB (21): PRINT AC(I,2);
4370  HTAB (31): PRINT AC(I,3)
```

```
4380  NEXT
4390  REM  PRINT NODAL SUPPRESSIONS
4400  PRINT : PRINT
4410  PRINT "LINE #  NODE #"; TAB( 3);
4420  PRINT "X    Y    Z    ";
4430  PRINT "RX    RY    RZ"
4440  PRINT
4450  FOR I = 1 TO AH%
4460  HTAB (2): PRINT I;
4470  HTAB (8): PRINT AG%(I,0);
4480  HTAB (17): PRINT AG%(I,1);
4490  HTAB (21): PRINT AG%(I,2);
4500  HTAB (26): PRINT AG%(I,3);
4510  HTAB (30): PRINT AG%(I,4);
4520  HTAB (35): PRINT AG%(I,5);
4530  PRINT  TAB( 5);AG%(I,6)
4540  NEXT
4550  PRINT : PRINT
4560  REM  PRINT MATERIAL PROPERTIES
4570  PRINT "MATERIAL PROPERTIES"
4580  PRINT
4590  FOR I = 1 TO AF% STEP 3
4600 I1 = I + 1
4610 I2 = I + 2
4620  PRINT "LINE #";: HTAB (8): PRINT I,I1,I2
4630  PRINT "GROUP";: HTAB (8): PRINT I,I1,I2
4640  PRINT "ELMT";: HTAB (8): PRINT AK(I,0),AK(I1,0),AK(I2,0)
4650  PRINT "AREA";: HTAB (8): PRINT AK(I,1),AK(I1,1),AK(I2,1)
4660  PRINT "I-YY";: HTAB (8): PRINT AK(I,2),AK(I1,2),AK(I2,2)
4670  PRINT "I-ZZ";: HTAB (8): PRINT AK(I,3),AK(I1,3),AK(I2,3)
4680  PRINT "E";: HTAB (8): PRINT AK(I,4),AK(I1,4),AK(I2,4)
4690  PRINT "V";: HTAB (8): PRINT AK(I,5),AK(I1,5),AK(I2,5)
4700  PRINT "J";: HTAB (8): PRINT AK(I,6),AK(I1,6),AK(I2,6)
4710  PRINT "X'";: HTAB (8): PRINT AK(I,7),AK(I1,7),AK(I2,7)
4720  PRINT "Y'";: HTAB (8): PRINT AK(I,8),AK(I1,8),AK(I2,8)
4730  PRINT "Z'";: HTAB (8): PRINT AK(I,9),AK(I1,9),AK(I2,9)
4740  PRINT : PRINT
4750  NEXT
4760  REM  PRINT "LOADS"
4770  PRINT : PRINT "LOADS"
4780  PRINT "LINE #  NODE #"; TAB( 3);
4790  PRINT "PX        PY        PZ        ";
4800  PRINT "MX        MY        MZ"
4810  FOR I = 1 TO AI%
4820  HTAB (2): PRINT I;
4830  HTAB (8): PRINT AJ(I,0);
4840  HTAB (15): PRINT AJ(I,1);
4850  HTAB (21): PRINT AJ(I,2);
4860  HTAB (28): PRINT AJ(I,3);
4870  HTAB (35): PRINT AJ(I,4);
4880  PRINT  TAB( 12);AJ(I,5);
4890  PRINT  TAB( 12);AJ(I,6)
4900  NEXT
4905  PR# 1
4906  RETURN
4910  REM  ********************************************
4920  REM  COPY OPTION
4930  HOME : PRINT "INSERT FLOPPY CONTAINING DATA TO BE COPIED"
4940  PRINT : PRINT "HIT RETURN WHEN READY TO PROCEED"
4950  INPUT A$
```

```
4955  GOSUB 2380: REM  READ DATA
4960  HOME : PRINT "INSERT FLOPPY TO BE COPIED INTO"
4970  PRINT : PRINT "HIT RETURN WHEN READY TO PROCEED"
4980  INPUT A$
5000  GOSUB 1750: REM  SAVE DATA
5010  RETURN
5020  REM   ***********************************************
5030  GOSUB 2390: REM    READ DATA
5050  HOME : INVERSE : PRINT "CHANGE OPTION": PRINT : NORMAL
5055  PRINT : PRINT "THIS ARE THE OPTIONS AVAILABLE:"
5060  PRINT : PRINT  TAB( 5);"1. ELEMENT GENERATOR CARDS."
5070  PRINT  TAB( 5);"2. NPCDS."
5080  PRINT  TAB( 5);"3. SUPPRESSIONS."
5090  PRINT  TAB( 5);"4. MATERIAL AND SECTION PROP."
5100  PRINT  TAB( 5);"5. LOADS."
5110  PRINT  TAB( 5);"6. EXIT."
5120  PRINT : INPUT "ENTER OPTION NUMBER ? ";A%
5130  IF (A% < 1) OR (A% > 6) THEN  HOME : PRINT "WRONG OPTION, TRY AGAIN.
   ": GOTO 300
5140  IF A% = 1 THEN  GOSUB 5230
5150  IF A% = 2 THEN  GOSUB 5530
5160  IF A% = 3 THEN  GOSUB 5800
5170  IF A% = 4 THEN  GOSUB 6150
5180  IF A% = 5 THEN  GOSUB 6570
5182  IF A% <  > 6 THEN  GOTO 5050
5185  GOSUB 1750: REM SAVE DATA
5190  RETURN
5210  REM   **************************************************
5220  HOME
5230  HOME : INVERSE : PRINT "CHANGE ELEMENT GENERATOR CARDS": NORMAL
5240  PRINT : PRINT "THIS ARE THE OPTION AVAILABLE:"
5250  PRINT : PRINT  TAB( 5);"1. CHANGE A LINE."
5260  PRINT  TAB( 5);"2. ADD OR DELETE A LINE."
5270  PRINT  TAB( 5);"3. EXIT."
5280  PRINT : INPUT "ENTER OPTION NUMBER ? ";A%
5290  IF (A% < 1) OR (A% > 3) THEN  HOME : PRINT "WRONG OPTION, TRY AGAIN.
   ": GOTO 5240
5300  IF A% = 1 THEN  INPUT "ENTER NUMBER OF LINE TO BE CHANGED ? ";I1: GOTO
   5360
5310  IF A% = 3 THEN  GOTO 5530
5320  INPUT "DO YOU WANT TO 'ADD' OR 'DELETE' A LINE ? ";A$
5330  IF A$ = "ADD" THEN AE% = AE% + 1:I1 = AE%: GOTO 5360
5340  IF A$ = "DELETE" THEN  HOME :AE% = AE% - 1: PRINT "LAST LINE HAS BEE
   N DELETED.": GOTO 5230
5350  HOME : PRINT "WRONG KEYWORD HAS BEEN ENTERED, TRY AGAIN.": GOTO 5230

5360  HOME
5370  PRINT "ENTER DATA FOR ELMT CARD # ";I1
5380  PRINT
5390  VTAB (5): INPUT "1.  ELEMENT TYPE (OPTION #) ? ";AB%(I1,1)
5400  INPUT "2.  GROUP NUMBER.";AB%(I1,2)
5410  INPUT "3.  NUMBER OF ELEMENTS ?";AB%(I1,3)
5420  INPUT "4.  NODE #1.";AB%(I1,4)
5430  IF AB%(I1,3) = 1 GOTO 5450
5440  INPUT "5.  INCREMENT #1 ?";AB%(I1,5)
5450  INPUT "6.  NODE #2. ?";AB%(I1,6)
5460  IF AB%(I1,3) = 1 GOTO 5480
5470  INPUT "7.  INCREMENT #2 ?";AB%(I1,7)
5480  IF AB%(I1,1) <  > 3 GOTO 5530: REM #1
5490  INPUT "8.  NODE #3 ?";AB%(I1,8)
```

```
5500  IF AB%(I1,3) = 1 GOTO 5530
5510  INPUT "9.  INCREMENT #3 ?";AB%(I1,9)
5530  RETURN
5540  REM   ***********************************************
5550  HOME
5560  HTAB (8): INVERSE : PRINT "CHANGE NPCOS": NORMAL
5570  PRINT : PRINT "THIS ARE THE OPTION AVAILABLE:"
5580  PRINT : PRINT  TAB( 5);"1. CHANGE A LINE."
5590  PRINT  TAB( 5);"2. ADD OR DELETE A NODE."
5600  PRINT  TAB( 5);"3. EXIT."
5610  PRINT : INPUT "ENTER OPTION NUMBER ? ";A%
5620  IF (A% < 1) OR (A% > 3) THEN  HOME : PRINT "WRONG OPTION, TRY AGAIN.
    ": GOTO 5570
5630  IF A% = 1 THEN  HOME : INPUT "CHANGE COORDINATES OF NODE # ? ";I1: GOTO
    5730
5640  IF A% = 3 THEN  GOTO 5780
5650  INPUT "DO YOU WANT TO 'ADD' OR 'DELETE' A NODE ? ";A$
5660  IF A$ = "ADD" THEN AD% = AD% + 1:I1 = AD%: GOTO 5730
5670  IF (A$ <  > "DELETE") THEN  HOME : PRINT "WRONG KEYWORD HAS BEEN ENT
    ERED, TRY AGAIN.": GOTO 5560
5680  HOME : INPUT "ENTER NUMBER OF NODE TO BE DELETED ? ";I1
5690 AC(I1,1) = 0
5700 AC(I1,2) = 0
5710 AC(I1,3) = 0
5720  HOME : PRINT "NODE HAS BEEN DELETED.": PRINT : GOTO 5550
5730  HOME : PRINT "ENTER THE NPCO FOR NODE # ";I1
5740  VTAB (5): INPUT "X COORDINATE ?";AC(I1,1)
5750  INPUT "Y COORDINATE ?";AC(I1,2)
5760  INPUT "Z COORDINATE ?";AC(I1,3)
5770  HOME : GOTO 5550
5780  RETURN
5790  REM  *****************************************************
5800  REM   ************************************************
5810  HOME
5820  INVERSE : PRINT "CHANGE NODAL SUPPRESSIONS": NORMAL
5830  PRINT : PRINT "THIS ARE THE OPTION AVAILABLE:"
5840  PRINT : PRINT  TAB( 5);"1. CHANGE A LINE."
5850  PRINT  TAB( 5);"2. ADD OR DELETE A LINE."
5860  PRINT  TAB( 5);"3. EXIT."
5870  PRINT : INPUT "ENTER OPTION NUMBER ? ";A%
5880  IF (A% < 1) OR (A% > 3) THEN  HOME : PRINT "WRONG OPTION, TRY AGAIN.
    ": GOTO 5830
5890  IF A% = 1 THEN  INPUT "ENTER NUMBER OF LINE TO BE CHANGED ? ";I: GOTO
    5950
5900  IF A% = 3 THEN  GOTO 6130
5910  INPUT "DO YOU WANT TO 'ADD' OR 'DELETE' A LINE ? ";A$
5920  IF A$ = "ADD" THEN AH% = AH% + 1:I = AH%: GOTO 5950
5930  IF A$ = "DELETE" THEN  HOME :AH% = AH% - 1: PRINT "LAST LINE HAS BEE
    N DELETED.": GOTO 5820
5940  HOME : PRINT "WRONG KEYWORD HAS BEEN ENTERED, TRY AGAIN.": GOTO 5820

5950  HOME : PRINT "ENTER NODE NUMBER AND Y OR N TO THE QUESTIONS.": VTAB
    (3)
5960  PRINT
5970  PRINT "NODAL SUPPRESSIONS CARD # ... ";I
5980  VTAB (6)
5990  INPUT "NODE NUMBER ? ";AG%(I,0)
5992  FOR J = 1 TO 6
5994 AG%(I,J) = 0
5996  NEXT J
```

```
6000  INPUT "SUPPRESS X ? ";I$
6010  IF (I$ = "Y") THEN AG%(I,1) = 1
6020  INPUT "SUPPRESS Y ? ";I$
6030  IF (I$ = "Y") THEN AG%(I,2) = 1
6040  INPUT "SUPPRESS Z ? ";I$
6050  IF (I$ = "Y") THEN AG%(I,3) = 1
6060  IF I1% = 1 THEN AG%(I,4) = 1:AG%(I,5) = 1:AG%(I,6) = 1: GOTO 6130
6070  INPUT "SUPPRESS THETA X ? ";I$
6080  IF (I$ = "Y") THEN AG%(I,4) = 1
6090  INPUT "SUPPRESS THETA Y ? ";I$
6100  IF (I$ = "Y") THEN AG%(I,5) = 1
6110  INPUT "SUPPRESS THETA Z ? ";I$
6120  IF (I$ = "Y") THEN AG%(I,6) = 1
6125  GOTO 5810
6130  RETURN
6140  REM    ***********************************************
6150  REM    ***************************************************
6160  HOME
6170  INVERSE : PRINT "CHANGE MATERIAL & SECTION PROP.": NORMAL
6180  PRINT : PRINT "THIS ARE THE OPTION AVAILABLE:"
6190  PRINT : PRINT  TAB( 5);"1. CHANGE A LINE."
6200  PRINT  TAB( 5);"2. ADD OR DELETE A LINE."
6210  PRINT  TAB( 5);"3. EXIT."
6220  PRINT : INPUT "ENTER OPTION NUMBER ? ";A%
6230  IF (A% < 1) OR (A% > 3) THEN  HOME : PRINT "WRONG OPTION, TRY AGAIN.
     ": GOTO 6180
6240  IF A% = 1 THEN  INPUT "ENTER NUMBER OF LINE TO BE CHANGED ? ";I: GOTO
     6310
6250  IF A% = 3 THEN  GOTO 6550
6260  INPUT "DO YOU WANT TO 'ADD' OR 'DELETE' A LINE ? ";A$
6270  IF A$ = "ADD" THEN AF% = AF% + 1:I = AF%: GOTO 6310
6280  IF A$ = "DELETE" THEN  HOME :AF% = AF% - 1: PRINT "LAST LINE HAS BEE
     N DELETED.": GOTO 6170
6290  HOME : PRINT "WRONG KEYWORD HAS BEEN ENTERED, TRY AGAIN.": GOTO 6170

6300  HOME
6310  HOME
6320  VTAB (3): PRINT "ENTER OPTION NUMBER"
6330  VTAB (7): PRINT  TAB( 3);"1. ONLY ROD ELEMENTS ARE USED."
6340  PRINT  TAB( 3);"2.  A MIXTURE OF RODS AND BEAM ELEMENTS ARE USED."
6350  VTAB (11)
6360  INPUT "ENTER OPTION NUMBER ?";I1%
6370  IF (I1% < 1) OR (I1% > 2) THEN  HOME : PRINT "WRONG OPTION NUMBER, T
     RY AGAIN ": VTAB (3): GOTO 6310
6380  HOME
6390  PRINT "MAT. & SECT. PROPERTIES FOR GROUP # ";I
6400  IF I1% = 1 THEN AK(I,0) = 1: GOTO 6420
6410  INPUT "ELEMENT TYPE ? ";AK(I,0)
6420  VTAB (3): INPUT "CROSS-SECTIONAL AREA ? ";AK(I,1)
6430  IF I1% = 1 GOTO 6470
6440  PRINT "MOMENTS OF INERTIA ABOUT LOCAL AXIS": PRINT
6450  INPUT "I (Y'-Y') ? ";AK(I,2)
6460  INPUT "I (Z'-Z') ? ";AK(I,3)
6470  PRINT : INPUT "MODULUS OF ELASTICITY E ? ";AK(I,4)
6480  IF I1% = 1 GOTO 6160
6490  INPUT "POISSON'S RATIO V ? ";AK(I,5)
6500  INPUT "POLAR MOMENT OF INERTIA J ? ";AK(I,6)
6510  INPUT "X' ? ";AK(I,7)
6520  INPUT "Y' ? ";AK(I,8)
6530  INPUT "Z' ? ";AK(I,9)
```

```
6540  GOTO 6160
6550  RETURN
6560  REM   **********************************
6570  REM   *****************************************************
6580  HOME
6590  INVERSE : PRINT "CHANGE LOAD CARDS" : NORMAL
6600  PRINT : PRINT "THIS ARE THE OPTION AVAILABLE:"
6610  PRINT : PRINT  TAB( 5);"1. CHANGE A LINE."
6620  PRINT  TAB( 5);"2. ADD OR DELETE A LINE."
6630  PRINT  TAB( 5);"3. EXIT."
6640  PRINT : INPUT "ENTER OPTION NUMBER ? ";A%
6650  IF (A% < 1) OR (A% > 3) THEN  HOME : PRINT "WRONG OPTION, TRY AGAIN.
     ": GOTO 6600
6660  IF A% = 1 THEN  INPUT "ENTER NUMBER OF LINE TO BE CHANGED ? ";I: GOTO
     6720
6670  IF A% = 3 THEN  GOTO 6900
6680  INPUT "DO YOU WANT TO 'ADD' OR 'DELETE' A LINE ? ";A$
6690  IF A$ = "ADD" THEN AI% = AI% + 1:I = AI%: GOTO 6720
6700  IF A$ = "DELETE" THEN  HOME :AI% = AI% - 1: PRINT "LAST LINE HAS BEE
     N DELETED.": GOTO 6590
6710  HOME : PRINT "WRONG KEYWORD HAS BEEN ENTERED, TRY AGAIN." : GOTO 6590

6720  HOME
6730  VTAB (3): PRINT "LOAD OPTIONS. "
6740  VTAB (7): PRINT  TAB( 3);"1. ONLY ROD ELEMENTS ARE USED."
6750  PRINT  TAB( 3);"2.  A MIXTURE OF RODS AND BEAM ELEMENTES ARE USED.

6760  VTAB (11): INPUT "ENTER LOAD OPTION ? ";I1%
6770  IF (I1% < 1) OR (I1% > 2) THEN  HOME : PRINT "WRONG OPTION NUMBER; T
     RY AGAIN ": GOTO 6730
6780  HOME
6790  PRINT "LOAD FOR LOAD CARD # ";I
6800  VTAB (6)
6810  INPUT "ENTER NODE NUMBER ? ";AJ(I,0)
6820  INPUT "ENTER F-X ? ";AJ(I,1)
6830  INPUT "ENTER F-Y ? ";AJ(I,2)
6840  INPUT "ENTER F-Z ? ";AJ(I,3)
6850  IF (I1% = 1) THEN  HOME : GOTO 1720
6860  INPUT "ENTER M-X ? ";AJ(I,4)
6870  INPUT "ENTER M-Y ? ";AJ(I,5)
6880  INPUT "ENTER M-Z ? ";AJ(I,6)
6890  GOTO 6580
6900  RETURN
6910  REM   ***********************************************
```

```
5  HOME : CLEAR
10  HTAB (10): PRINT "PROGRAM ASTRA1": PRINT
20  REM   25 NOVEMBER 1980
60  HTAB (5): PRINT "ANALISING STRUCTURES WITH APPLE"
80  GOSUB 1710: REM   ALLOCATE ARRAYS
90  GOSUB 2150: REM  READ DATA
110  GOSUB 5780: REM  PRINT INPUT DATA
155  GOSUB 8610: REM  CALCULATE BANDWIDTH
158  REM GOSUB 8750: REM STORE BANDWIDTH CF
160  GOSUB 7840: REM   DELETE OLD SSM AND CREATE A NEW NULL (ALL 0) SSM RE
     GISTER
200 BP = AD% * 6: REM    SIZE OF SSM BP*BP
210  PRINT "SIZE OF SSM IS ";BP
220  FOR I8 = 1 TO AE%: REM  # ELEMENT CARD LOOP
230  GOSUB 3530: REM   SECT PROP.
231  FOR I9 = 1 TO AB%(I8,3): REM  ELEMT/CARD LOOP
233  GOSUB 250: REM  CALCULATE STRUCTURE SSM
234  NEXT I9
235  NEXT I8
236  PRINT : PRINT "ASSY OF SSM IS COMPLETE": PRINT
237  GOSUB 7310: REM  ASSY OF SS IN DOFIU(I,2)
238  PRINT "SS HAS BEEN STORED IN DOFIU"
239  GOSUB 7420: REM  STORE LOAD IN DOFIU(I,1)
240  PRINT "P HAVE BEEN STORED IN DOFIU"
241  GOSUB 8380: REM  SAVE DOFIU IN DISK
242  REM GOSUB 8060:REM PRINT DOFIU
243  HOME
244  PRINT "PROGRAM ASTRA1 IS COMPLETED.  TO CONTINUE LOAD AND RUN PROGRAM
      ASTRA2"
245  END
246  REM   ****************************************************
248  REM  CALCULATION OF SSM
250 BR% = BR% + 1: REM  ELEMENT COUNTER
260  GOSUB 1860: REM     CK FOR VERTICAL ELEMENTS,OBTAIN N1,N2,NODAL COORD
     S., DIRECTION COSINES.
265  PRINT : PRINT "ELEMENT # ";BR%; SPC( 2);"NODE I= ";N1; SPC( 2);" NODE
     J= ";N2
270  GOSUB 2040: REM  ACTIVATE D.O.F. IN USE
280  PRINT : PRINT "THE DIRECTION COSINES CX,CY,CZ FOR ELEMENT ";BR%;" ARE
      "

290  PRINT CX; SPC( 3);CY; SPC( 3);CZ
295  IF (AB%(I8,1) = 2) THEN  GOSUB 4004: REM   CALCULATE CG,SG
300  GOSUB 3240: REM  CLEAR MATRICES
310  REM  CALCULATE TRANSFORMATION MATRIX
320  IF (AB%(I8,1) = 1) AND (BI% = 1) THEN  GOSUB 5170: REM  VERTICAL ROD

330  IF (AB%(I8,1) = 1) AND (BI% = 0) THEN  GOSUB 5010: REM   NON-VERTICAL
      ROD
340  IF (AB%(I8,1) = 2) AND (BI% = 1) THEN  GOSUB 4210: REM     VERTICAL B
     EAM
350  IF (AB%(I8,1) = 2) AND (BI% = 0) THEN  GOSUB 4520: REM    NONVERTICAL
      BEAM
360  GOSUB 7660: REM  STORE ETM
370  REM  GOSUB 6870: REM  PRINT TRANS  MATRIX
380  GOSUB 5260: REM  FIND INVERSE (TRANSPOSE) OF TRANSFORMATION MATRIX.
390  REM  GOSUB 6960: REM   PRINT TRANSPOSE OF ELMT TRANS MATRIX
400  REM  ELEMENT STIFFNESS MATRIX (LOCAL)
```

```
410  IF AB%(I8,1) = 1 THEN  GOSUB 5340: REM  SPACE TRUSS
420  IF AB%(I8,1) = 2 THEN  GOSUB 2870: REM  SPACE BEAM
430  PRINT
440  IF AB%(I8,1) = 1 THEN  GOSUB 8190: REM STORE ESM ROD
445  IF AB%(I8,1) = 2 THEN  GOSUB 8800: REM STORE ESM BEAM
450  REM  GOSUB 7050: REM  PRINT STIFF MATRIX
460  GOSUB 5410: REM    ELEMENT STIFFNESS MATRIX (GLOBAL)
470  REM  GOSUB 7130: REM  PRINT ELMT STIFF MATRIX
480  REM    ***********************************************************
     ********************************************
490  REM  ADD ELEMENTS STIFFNESS TO STRUCTURE STIFFNESS MATRIX.
500  PRINT "ASSY OF STRUCTURE STIFFNESS MATRIX
510  REM    ***********************************************************
     ********************************************
520  REM  R2% IS THE ADDRESS WHERE THE VALUE IN SSM WILL BE STORED
530 D$ =  CHR$ (4)
535 A$ = "SSM1X"
540  PRINT D$;"OPEN ";A$;", L20"
550  FOR I = 1 TO 12
560  FOR J = 1 TO I
565 BQ = 0
570  IF BH(I,J) = 0 GOTO 810
580  GOSUB 8700: REM  CALCULATE ADDRESS OF SSM IN DISK
680  PRINT D$;"READ ";A$;", R";R2%
690  INPUT BQ
700 BQ = BQ + BH(I,J)
780  PRINT D$;"WRITE ";A$;", R";R2%
790  PRINT BQ
810  NEXT J
820  NEXT I
830  PRINT D$;"CLOSE ";A$
1540  PRINT "ELEMENT # ";BR%;" STIFFNESS HAS BEEN ADDED TO SSM"
1545  RETURN
1700  REM    ***********************************************************
     ***************************************************************
     ******************
1710  REM   ALLOCATION OF ARRAYS I
1720  DIM AAID$(1),ABEGEN%(25,9)
1730  DIM ACNPCO(50,3)
1740  DIM AGNDOF%(50,6)
1750  DIM AJPNODE(50,6)
1760  DIM AKMSPROP(10,9)
1770  DIM DOFIU(300,3)
1780  DIM BB(12,12): REM  TRANS. MATRIX
1790  DIM BC(12,12): REM   STIFF. MATRIX (ELEMENT-LOCAL)
1800  DIM BF(12,12): REM   TRANS. MATRIX INVERSE (TRANSPOSE)
1810  DIM BH(12,12): REM  STIFF. MATRIX (ELEMENT-GLOBAL)
1820  DIM BT(12,12): REM  BT=BF*BC USED TO CALCULATE ESM (GLOBAL)
1830  DIM BM(3,3),BN(3,3),BO(3,3),DX(3),EX(3): REM  USED TO CALCULATE ALPH
     A
1840  RETURN
1850  REM    ********************************************
1860  REM  CK FOR VERTICAL ELEMENTS, OBTAIN N1,N2,NFCO,DIRECTION COSINES
1870 BI% = 0
1880 N1 = ABX(I8,4) + (ABX(I8,5) * (I9 - 1))
1890 N2 = ABX(I8,6) + (ABX(I8,7) * (I9 - 1))
1900  IF N1 > N2 THEN N3 = N2:N2 = N1:N1 = N3:N3 = 0
1920 X1 = AC(N1,1)
1930 Y1 = AC(N1,2)
1940 Z1 = AC(N1,3)
```

```
1950 X2 = AC(N2,1)
1960 Y2 = AC(N2,2)
1970 Z2 = AC(N2,3)
1980  REM  CK FOR VERTICAL ELEMENTS
1990 BI% = 0
2000  IF ( ABS (X2 - X1) > .00001) GOTO 2010
2005  IF ( ABS (Z2 - Z1) > .00001) GOTO 2010
2006 BI% = 1
2010  REM  CALCULATE DIRECTION COSINES
2012 NL = ((X2 - X1) ↑ 2 + (Y2 - Y1) ↑ 2 + (Z2 - Z1) ↑ 2) ↑ .5
2013 CX = (X2 - X1) / NL
2014 CY = (Y2 - Y1) / NL
2015 CZ = (Z2 - Z1) / NL
2020  RETURN
2030  REM  ****************************************************
2040  REM  ACTIVATING N.D.F. USED
2050 K = (N1 - 1) * 6
2060 K3 = (N2 - 1) * 6
2070 K2 = 3
2080  IF AB%(IB,1) = 2 THEN K2 = 6
2090  FOR I = 1 TO K2
2100 DOFIU((K + I),0) = 1
2110 DOFIU((K3 + I),0) = 1
2120  NEXT I
2130  RETURN
2140  REM  ****************************************************
2150  REM  SUBROUTINE READ DATA
2160 D$ =  CHR$ (4): REM  CTRL D
2161  PRINT : INPUT "ENTER DRAWING CODE NUMBER ? ";AA$(0)
2162  PRINT D$;"OPEN ID"
2163  PRINT D$;"DELETE ID"
2164  PRINT D$;"OPEN ID, L12"
2165  PRINT D$;"WRITE ID,R1"
2166  PRINT AA$(0)
2167  PRINT D$;"CLOSE ID"
2180  PRINT : PRINT "READING INPUT INFORMATION"
2190  PRINT D$;"OPEN ";AA$(0);", L12"
2200  PRINT D$;"READ ";AA$(0);",R";1
2210  INPUT AD%
2220  PRINT D$;"READ ";AA$(0);",R";2
2230  INPUT AE%
2240  PRINT D$;"READ ";AA$(0);",R";3
2250  INPUT AF%
2260  PRINT D$;"READ ";AA$(0);",R";4
2270  INPUT AH%
2280  PRINT D$;"READ ";AA$(0);",R";5
2290  INPUT AI%
2300  REM  READ ELEMENT CARDS
2310  FOR I = 1 TO AE%
2320 J = ((I - 1) * 9) + 10
2330  FOR K = 1 TO 9
2340  PRINT D$;"READ ";AA$(0);",R";(J + K)
2350  INPUT AB%(I,K)
2360  NEXT K
2370  NEXT I
2380  REM  READ NPCO
2390 A = (AE% * 9) + 10
2400  FOR I = 1 TO AD%
2410 J = ((I - 1) * 3) + A
2420  FOR K = 1 TO 3
```

```
2430  PRINT D$;"READ ";AA$(0);",R";(J + K)
2440  INPUT AC(I,K)
2450  NEXT K
2460  NEXT I
2470  REM   READ SUPPRESSIONS
2480 A = (AE% * 9) + (AD% * 3) + 10
2490  FOR I = 1 TO AH%
2500 J = A + ((I - 1) * 7)
2510  FOR K = 0 TO 6
2520  PRINT D$;"READ ";AA$(0);",R";(J + 1 + K)
2530  INPUT AG%(I,K)
2540  NEXT K
2550  NEXT I
2560  REM  READ MAT. & SECT. PROP.
2570 A = A + (AH% * 7)
2580  FOR I = 1 TO AF%
2590 J = A + ((I - 1) * 10)
2600  FOR K = 0 TO 9
2610  PRINT D$;"READ ";AA$(0);",R";(J + 1 + K)
2620  INPUT AK(I,K)
2630  NEXT K
2640  NEXT I
2650  REM  READ LOADS
2660 A = A + (AF% * 10)
2670  FOR I = 1 TO AI%
2680 J = A + ((I - 1) * 7)
2690  FOR K = 0 TO 6
2700  PRINT D$;"READ ";AA$(0);",R";(J + 1 + K)
2710  INPUT AJ(I,K)
2720  NEXT K
2730  NEXT I
2740  PRINT D$;"CLOSE";AA$(0)
2750  RETURN
2760  REM   **********************************
2770  REM  READ HEADING
2780 D$ =  CHR$ (4)
2790  PRINT D$;"OPEN HEADING, L20"
2800  FOR I = 0 TO 5
2810  PRINT D$;"READ HEADING,R";I
2820  INPUT AA$(I)
2830  NEXT
2840  PRINT D$;"CLOSE HEADING"
2850  RETURN
2860  REM   ***********************************************************
2870  REM   BEAM STIFFNESS MATRIX
2880 BC(1,1) = (BE * BA) / NL
2890 BC(7,7) = BC(1,1)
2900 BC(7,1) =  - BC(1,1)
2910 BC(2,2) = (12 * BE * IZ) / (NL ↑ 3)
2920 BC(8,8) = BC(2,2)
2930 BC(8,2) =  - BC(2,2)
2940 BC(3,3) = (12 * BE * IY) / (NL ↑ 3)
2950 BC(9,9) = BC(3,3)
2960 BC(9,3) =  - BC(3,3)
2970 BC(4,4) = (BG * BJ) / NL
2980 BC(10,10) = BC(4,4)
2990 BC(10,4) =  - BC(4,4)
3000 BC(5,5) = (4 * BE * IY) / NL
3010 BC(11,11) = BC(5,5)
3020 BC(6,6) = (4 * BE * IZ) / NL
```

```
3030 BC(12,12) = BC(6,6)
3040 BC(5,3) =  - (6 * BE * IY) / (NL ↑ 2)
3050 BC(11,9) =  - BC(5,3)
3060 BC(9,5) =  - BC(5,3)
3070 BC(11,3) = BC(5,3)
3080 BC(6,2) = (6 * BE * IZ) / (NL ↑ 2)
3090 BC(8,6) =  - BC(6,2)
3100 BC(12,8) =  - BC(6,2)
3110 BC(12,2) = BC(6,2)
3120 BC(11,5) = BC(5,5) / 2
3130 BC(12,6) = BC(6,6) / 2
3140 BC(3,5) = BC(5,3)
3142 BC(2,6) = BC(6,2)
3144 BC(1,7) = BC(7,1)
3146 BC(2,8) = BC(8,2)
3148 BC(3,9) = BC(9,3)
3150 BC(6,8) = BC(8,6)
3152 BC(5,9) = BC(9,5)
3154 BC(4,10) = BC(10,4)
3156 BC(3,11) = BC(11,3)
3158 BC(2,12) = BC(12,2)
3160 BC(5,11) = BC(11,5)
3162 BC(6,12) = BC(12,6)
3164 BC(8,12) = BC(12,8)
3166 BC(9,11) = BC(11,9)
3210  PRINT : PRINT "LOCAL STIFF. OF ELMT ";BR%;" IS COMPLETED"
3220  RETURN
3230  REM   ****************************************
3240  REM  SUBROUTINE CLEAR MATRICES
3250  PRINT "CLEARING BB,BC,BF,BH"
3260  FOR K1 = 1 TO 12
3270  FOR K2 = 1 TO 12
3280 BB(K1,K2) = 0
3290 BC(K1,K2) = 0
3300 BF(K1,K2) = 0
3310 BH(K1,K2) = 0
3320 BT(K1,K2) = 0
3330  NEXT K2
3340  NEXT K1
3350  FOR I = 1 TO 3
3360  FOR J = 1 TO 3
3370 BN(I,J) = 0
3380 BO(I,J) = 0
3390  NEXT J
3400  NEXT I
3410  RETURN
3420  REM   ****************************************
3430  REM  MATRIX MULTIPLICATION
3440  REM   C=AB (BOTH MATRICES ARE 12*12)
3450  FOR K1 = 1 TO 12
3460  FOR K2 = 1 TO 12
3470  FOR K3 = 1 TO 12
3480 C(K1,K2) = C(K1,K2) + A(K1,K3) * B(K3,K2)
3490  NEXT K3
3500  NEXT K2
3510  NEXT K1
3520  REM   ****************************************
3530  REM  SECT. PROP. FOR THE GROUPS
3540  IF (AB%(I8,2) = NG%) GOTO 3730
3550 NG% = AB%(I8,2)
```

```
3560  GOSUB 3750: REM  CLEAR SECT. PROP.
3570 NT% = AK(NG%,0)
3580 BA = AK(NG%,1)
3590 BE = AK(NG%,4)
3600  IF (AB%(I8,1) = 1) GOTO 3730
3610 IY = AK(NG%,2)
3620 IZ = AK(NG%,3)
3630 BD = AK(NG%,5)
3640 BJ = AK(NG%,6)
3650 BX = AK(NG%,7)
3660 BY = AK(NG%,8)
3670 BZ = AK(NG%,9)
3680 BG = BE / (2 * (1 + BD))
3730  RETURN
3740  REM   ******************************************
3750  REM  CLEAR SECTION PROPERTIES
3760 NT% = 0
3770 BA = 0
3780 BE = 0
3790 IY = 0
3800 IZ = 0
3810 BD = 0
3820 BJ = 0
3830 BG = 0
3840 BX = 0
3850 BY = 0
3860 BZ = 0
3870 BG = 0
3930  RETURN
4002  REM   ********************************************
4004  REM  CAL ANGLE OF ROTATION GAMMA ABOUT X AXIS
4005  FOR I = 1 TO 3
4006  FOR J = 1 TO 3
4008 BB(I,J) = 0
4009  NEXT J
4010  NEXT I
4012 CG = 0:SG = 0
4014  FOR I = 1 TO 3
4016 DX(I) = 0:EX(I) = 0
4018  NEXT I
4021  IF BI% < > 1 GOTO 4030: REM  CK FOR VERTICAL BEAMS
4022  GOSUB 5170: REM  ETM FOR VERTICAL TRUSS
4028  GOTO 4040
4030  GOSUB 4810: REM  ROT MATRIX R-A*R-B
4040  REM  CALCULATE X-FS
4050 DX(1) = BX - X1
4060 DX(2) = BY - Y1
4070 DX(3) = BZ - Z1
4075  IF (BI% = 1) GOTO 4182: REM  VERTICAL BEAM
4080  REM  CALCULATE X-PG
4086  REM  CG AND SG FOR NON-VERTICAL BEAMS
4090  REM  MATRIX MULTIPLICATION X-PG=BO*X-FS
4100  FOR K1 = 1 TO 3
4110  FOR K2 = 1 TO 3
4120 EX(K1) = EX(K1) + BB(K1,K2) * DX(K2)
4130  NEXT K2
4140  NEXT K1
4150  REM   CALCULATE COS AND SIN OF GAMMA
4160 EX(0) = (EX(2) ↑ 2 + EX(3) ↑ 2) ↑ .5
4170 CG = EX(2) / EX(0)
```

```
4180 SG = EX(3) / EX(0)
4181  GOTO 4189
4182  REM   GC,SG FOR VERTICAL BEAMS
4184 BK = (DX(1) ↑ 2 + DX(3) ↑ 2) ↑ .5
4186 SG = DX(3) / BK
4188 CG =  - (DX(1) * CY) / BK
4189  PRINT : PRINT "THE FOLLOWING ARE THE DIRECTION COSINES CG,SG ";CG; SPC(
     3);SG: PRINT
4190  RETURN
4200  REM   ******************************************
4210  REM  TRASFORMATION MATRIX FOR VERTICAL BEAMS
4220 BB(1,2) = CY
4225 BB(2,1) =  - CY * CG
4230 BB(2,3) = SG
4235 BB(3,1) = CY * SG
4240 BB(3,3) = CG
4245  GOSUB 4350: REM  FILL UP MATRIX
4250  RETURN
4345  REM   **********************************
4350  REM  FILL UP BB MATRIX FOR BEAM
4360  FOR I = 1 TO 3
4370  FOR J = 1 TO 3
4380 K = I + 3:L = J + 3
4385 M = I + 6:N = J + 6
4390 O = I + 9:P = J + 9
4400 BB(K,L) = BB(I,J)
4402 BB(M,N) = BB(I,J)
4404 BB(O,P) = BB(I,J)
4410  NEXT J
4420  NEXT I
4500  RETURN
4510  REM   ********************************************
4520  REM  TRANSFORMATION MATRIX FOR NON-VERTICAL BEAMS
4530 BK = (CX ↑ 2 + CZ ↑ 2) ↑ .5
4540 BB(1,1) = CX
4560 BB(1,2) = CY
4580 BB(1,3) = CZ
4600 BB(2,1) = (( - CX * CY * CG) - (CZ * SG)) / BK
4620 BB(2,2) = BK * CG
4640 BB(2,3) = (( - CY * CZ * CG) + (CX * SG)) / BK
4660 BB(3,1) = ((CX * CY * SG) - (CZ * CG)) / BK
4680 BB(3,2) =  - BK * SG
4700 BB(3,3) = ((CY * CZ * SG) + (CX * CG)) / BK
4710  GOSUB 4350: REM  FILL UP BB MATRIX
4790  RETURN
4800  REM   ******************************************
4810  REM  ROTATION MATRIX R-ALPHA*R-BETA
4820 BK = (CX ↑ 2 + CZ ↑ 2) ↑ .5
4900 BB(1,1) = CX
4910 BB(1,2) = CY
4920 BB(1,3) = CZ
4930 BB(2,1) =  - (CX * CY) / BK
4940 BB(2,2) = BK
4950 BB(2,3) =  - (CY * CZ) / BK
4960 BB(3,1) =  - CZ / BK
4970 BB(3,2) = 0
4980 BB(3,3) = CX / BK
4990  RETURN
5000  REM   ********************************************
5010  REM   TRANSFORMATION MATRIX FOR NON-VERTICAL SPACE TRUSS
```

```
5020  GOSUB 4810: REM  ROT MATRIX R-A*R-B
5030  FOR I = 1 TO 3
5040  FOR J = 1 TO 3
5100 K = I + 6
5110 L = J + 6
5120 BB(K,L) = BB(I,J)
5130  NEXT J
5140  NEXT I
5150  RETURN
5160  REM   *********************************************
5170  REM  TRANSFORMATION MATRIX FOR VERTICAL SPACE TRUSS
5180 BB(1,2) = CY
5190 BB(7,8) = CY
5200 BB(2,1) =  - CY
5210 BB(8,7) =  - CY
5220 BB(3,3) = 1
5230 BB(9,9) = 1
5240  RETURN
5250  REM   ************************************************************
5260  REM  FIND INVERSE OF TRANSFORMATION MATRIX R
5270  FOR K1 = 1 TO 12
5280  FOR K2 = 1 TO 12
5290 BF(K2,K1) = BB(K1,K2)
5300  NEXT K2
5310  NEXT K1
5320  RETURN
5330  REM   ****************************************************
5340  REM  STIFFNESS MATRIX FOR ROD
5350 BC(1,1) = (BE * BA) / RL
5360 BC(7,1) =  - BC(1,1)
5370 BC(1,7) =  - BC(1,1)
5380 BC(7,7) = BC(1,1)
5390  RETURN
5400  REM   *********************************************
5410  REM  ELEMENT STIFFNESS MATRIX (GLOBAL)
5420  PRINT "CALCULATION OF ELEMT. STIFF MATRIX (GLOBAL)"
5430  REM    BH=BF*BC*BB
5435  FOR A = 1 TO 12 STEP 3
5436 B = A + 2
5440  FOR K1 = A TO B
5441  FOR K2 = 1 TO 12
5442  FOR K3 = A TO B
5443 BT(K1,K2) = BT(K1,K2) + (BF(K1,K3) * BC(K3,K2))
5444  NEXT K3
5445  NEXT K2
5446  NEXT K1
5447  NEXT A
5505  FOR A = 1 TO 12 STEP 3
5506 B = A + 2
5510  FOR K1 = 1 TO 12
5520  FOR K2 = A TO B
5530  FOR K3 = A TO B
5540 BH(K1,K2) = BH(K1,K2) + (BT(K1,K3) * BB(K3,K2))
5550  NEXT K3
5560  NEXT K2
5570  NEXT K1
5575  NEXT A
5580  RETURN
5590  REM   *********************************************
5600  REM  READ STRUCTURE SUPPRESSIONS INTO SS
```

```
5610 D$ =  CHR$ (4): REM   CTRL D
5620  PRINT D$;"OPEN ";AA$(0);", L6"
5630 A = (AE% * 9) + (AD% * 3) + 10
5640  FOR I = 1 TO AH%
5650 J = A + ((I - 1) * 7)
5660 K = 0
5670  PRINT D$;"READ ";AA$(0);",R";(J + 1 + K)
5680  INPUT N
5690  FOR K = 1 TO 6
5700  PRINT D$;"READ ";AA$(0);",R";(J + 1 + K)
5710  INPUT K1
5720  IF K1 > 0 THEN SS(N1) = 2
5730  NEXT K
5740  NEXT I
5750  PRINT D$;"CLOSE";AA$(0)
5760  RETURN
5770  REM  ************************************************************
5780  HOME : PRINT "LIST INPUT INFORMATION": PRINT
5980  REM   PRINT ELEMENTS
6000  PRINT "ELEMENT CARDS": PRINT
6010  PRINT "  LINE";
6020  HTAB (8): PRINT "ELMT  GROUP  # OF";
6040  HTAB (27): PRINT "NODE  INC  NODE  INC"
6050  PRINT "    #";
6060  PRINT  SPC( 3);"TYPE"; SPC( 4);"#"; SPC( 4);"ELMT"; SPC( 2);
6070  PRINT "1"; SPC( 5);"1"; SPC( 4);"2"; SPC( 5);"2": PRINT
6090  FOR I = 1 TO AE%
6100  HTAB (4): PRINT I;
6110  HTAB (9): PRINT AB%(I,1);
6120  HTAB (16): PRINT AB%(I,2);
6130  HTAB (22): PRINT AB%(I,3);
6140  HTAB (26): PRINT AB%(I,4);
6150  HTAB (33): PRINT AB%(I,5);
6160  HTAB (39): PRINT AB%(I,6);
6170  HTAB (1): PRINT  TAB( 6);AB%(I,7)
6200  NEXT
6210  REM  PRINT NPCO
6220  PRINT : PRINT
6230  PRINT "NODE #"; SPC( 3);"X-COORD"; SPC( 4);"Y-COORD"; SPC( 4);"Z-COO
     RD"
6240  PRINT
6250  FOR I = 1 TO AD%
6260  PRINT I;
6270  HTAB (11): PRINT AC(I,1);
6280  HTAB (21): PRINT AC(I,2);
6290  HTAB (31): PRINT AC(I,3)
6300  NEXT
6310  REM  PRINT NODAL SUPPRESSIONS
6320  PRINT : PRINT
6330  PRINT "LINE #  NODE #"; SPC( 3);
6340  PRINT "X   Y   Z   ";
6350  PRINT "RX   RY   RZ"
6360  PRINT
6370  FOR I = 1 TO AH%
6380  HTAB (3): PRINT I;
6390  HTAB (10): PRINT AG%(I,0);
6400  HTAB (17): PRINT AG%(I,1);
6410  HTAB (21): PRINT AG%(I,2);
6420  HTAB (26): PRINT AG%(I,3);
6430  HTAB (30): PRINT AG%(I,4);
```

```
6440  HTAB (35): PRINT AG%(I,5);
6450  PRINT  SPC( 4);AG%(I,6)
6460  NEXT
6470  PRINT : PRINT
6480  REM  PRINT MATERIAL PROPERTIES
6490  PRINT "MATERIAL PROPERTIES"
6500  PRINT
6510  FOR I = 1 TO AF% STEP 3
6520 I1 = I + 1
6530 I2 = I + 2
6540  PRINT "LINE #";: HTAB (8): PRINT I,I1,I2
6550  PRINT "GROUP";: HTAB (8): PRINT I,I1,I2
6560  PRINT "ELMT";: HTAB (8): PRINT AK(I,0),AK(I1,0),AK(I2,0)
6570  PRINT "AREA";: HTAB (8): PRINT AK(I,1),AK(I1,1),AK(I2,1)
6580  PRINT "I-YY";: HTAB (8): PRINT AK(I,2),AK(I1,2),AK(I2,2)
6590  PRINT "I-ZZ";: HTAB (8): PRINT AK(I,3),AK(I1,3),AK(I2,3)
6600  PRINT "E";: HTAB (8): PRINT AK(I,4),AK(I1,4),AK(I2,4)
6610  PRINT "V";: HTAB (8): PRINT AK(I,5),AK(I1,5),AK(I2,5)
6620  PRINT "J";: HTAB (8): PRINT AK(I,6),AK(I1,6),AK(I2,6)
6630  PRINT "X'";: HTAB (8): PRINT AK(I,7),AK(I1,7),AK(I2,7)
6640  PRINT "Y'";: HTAB (8): PRINT AK(I,8),AK(I1,8),AK(I2,8)
6650  PRINT "Z'";: HTAB (8): PRINT AK(I,9),AK(I1,9),AK(I2,9)
6660  PRINT : PRINT
6670  NEXT
6680  REM  PRINT "LOADS"
6690  PRINT : PRINT "LOADS"
6700  PRINT "LINE #  NODE #"; SPC( 3);
6710  PRINT "PX        PY        PZ        ";
6720  PRINT "MX        MY        MZ"
6730  FOR I = 1 TO AI%
6740  HTAB (3): PRINT I;
6750  HTAB (11): PRINT AJ(I,0);
6760  HTAB (17): PRINT AJ(I,1);
6770  HTAB (24): PRINT AJ(I,2);
6780  HTAB (32): PRINT AJ(I,3);
6790  HTAB (40): PRINT AJ(I,4);
6800  PRINT  TAB( 12);AJ(I,5);
6810  PRINT  TAB( 20);AJ(I,6)
6820  NEXT
6850  RETURN
6860  REM   ***********************************************
6870  REM  PRINT TRANSFORMATION MATRIX
6880  PRINT "TRANSFORMATION # ";BR%
6890  FOR I = 1 TO 12
6900  FOR J = 1 TO 12
6910  PRINT BE(I,J);"  ";
6920  NEXT J
6930  NEXT I
6940  RETURN
6950  REM   ****************************************
6960  REM   PRINT TRANSFORMATION MATRIX TRANSPOSE
6970  PRINT "TRANSFORMATION TRANSPOSE # ";BR%
6980  FOR I = 1 TO 12
6990  FOR J = 1 TO 12
7000  PRINT BF(I,J);"   ";
7010  NEXT J
7020  NEXT I
7030  RETURN
7040  REM    *********************          **********************
7050  PRINT "STIFFNESS MATRIX (LOCAL) # ";BR%
```

```
7060  FOR I = 1 TO 12
7070  FOR J = 1 TO 12
7080  PRINT BC(I,J);"  ";
7090  NEXT J
7100  NEXT I
7110  RETURN
7120  REM  ********************************************
7130  PRINT "STIFF MATRIX (GLOBAL) # ";BR%
7140  FOR I = 1 TO 12
7150  FOR J = 1 TO 12
7160  PRINT BH(I,J);"  ";
7170  NEXT J
7180  NEXT I
7190  RETURN
7200  REM   *****************************************************
7210  REM  REARANGE DOFIU IN # ORDER OF USAGE
7220 K = 0: REM  RECORD/NUMBER OF DOF IN USE
7230  FOR I = 1 TO 300
7240  IF (DOFIU(I,0) = 0) GOTO 7270
7250 K = K + 1
7260 DOFIU(K,3) = I
7270  NEXT I
7280 CC = K: REM  RANK OF CONSOLIDATED MATRIX K
7290  RETURN
7300  REM   ********************************************
7310  REM  SET AND STORE NODAL SUPPRESSIONS
7320  FOR I = 1 TO AH%
7330 R% = (AG%(I,0) - 1) * 6
7340  FOR J = 1 TO 6
7350  IF AG%(I,J) = 0 GOTO 7380
7360 R1% = R% + J
7370 DOFIU(R1%,2) = 1
7380  NEXT J
7390  NEXT I
7400  RETURN
7410  REM   ******************************************************
7420  REM   STORE NODAL LOADS IN STRUCTURE LOAD MATRIX
7430  FOR I = 1 TO AI%
7440  FOR J = 1 TO 6
7450  IF AJ(I,J) = 0 GOTO 7490
7460 R% = (AJ(I,0) - 1) * 6
7470 R1% = R% + J
7480 DOFIU(R1%,1) = AJ(I,J)
7490  NEXT J
7500  NEXT I
7510  RETURN
7520  REM   ************************************************************
     *
7660  REM  SAVE ETM (PER ELEMENT)
7670  REM  ONLY TOP-LEFT 3X3 MATRIX IS STORED
7680 D$ =  CHR$ (4)
7690 B$ =  STR$ (BR%)
7700 A$ = "ETM" + B$
7710 R2% = 0
7718  PRINT D$;"OPEN ";A$
7719  PRINT D$;"DELETE ";A$
7720  PRINT D$;"OPEN ";A$;", L20"
7730  FOR I = 1 TO 3
7740  FOR J = 1 TO 3
7765 R2% = R2% + 1
```

```
7770  PRINT D$;"WRITE ";A$;",R";R2%
7780  PRINT BB(I,J)
7790  NEXT J
7800  NEXT I
7810  PRINT D$;"CLOSE ";A$
7820  RETURN
7830  REM   **********************************************
7840  REM   CLEAR SSM
7850 D$ =  CHR$ (4)
7880 A$ = "SSM1X"
7890  PRINT D$;"OPEN ";A$
7900  PRINT D$;"DELETE ";A$
7910  PRINT "SSM HAVE BEEN CLEARED": PRINT
7912  REM  CREATE A NULL SSM
7913  PRINT "CREATING A NULL SSM": PRINT
7916 K3 = 0
7918 K1 = CI * (AD% * 6)
7919  PRINT K1,CG,AD%
7920  PRINT D$;"OPEN ";A$;", L20"
7924  FOR I = 1 TO K1
7927  PRINT D$;"WRITE ";A$;",R";I
7928  PRINT K3
7930  NEXT I
7931  PRINT D$;"CLOSE ";A$
7932  PRINT "LAST ADDRESS OF SSM IS ";K1
7933  PRINT : PRINT "NULL SSM HAS BEEN CREATED": PRINT
7939  RETURN
7940  REM  *******************************************
7950  REM  ERROR MESSAGES
7960  PRINT "R% < 0 IN ASS/ OF SSM": GOTO 7980
7970  PRINT "NUMBER OF NODES EXCEED 50 NODE LIMIT/# OF DOF EXCEED 300 LIMI
     T": GOTO 7980
7980  STOP
7990  REM   ***************************************************
8050  REM   *******************************************
8060  REM  PRINT DOFIU AND BU
8070  PRINT
8080  PRINT "CHECK PRINTOUT OF DOFIU AND BU"
8090  FOR I = 1 TO BF
8100  PRINT I; SPC( 2);
8110  FOR J = 0 TO 3
8120  PRINT DOFIU(I,J); SPC( 2);
8130  NEXT J
8140  PRINT
8150  NEXT I
8160  RETURN
8170  REM   *******************************************
8190  REM  STORE ESM ROD (LOCAL)
8210 D$ =  CHR$ (4)
8220 B$ =  STR$ (BR%)
8230 A$ = "ESM" + B$
8245  PRINT D$;"OPEN ";A$
8246  PRINT D$;"DELETE ";A$
8250  PRINT D$;"OPEN ";A$;", L20"
8295 R2% = 1
8300  PRINT D$;"WRITE ";A$;",R";R2%
8310  PRINT BC(1,1)
8340  PRINT D$;"CLOSE ";A$
8350  PRINT "ESM SAVED ON DISK"
8360  RETURN
```

```
8370  REM   **********************************************
8380  REM   SAVE DOFIU(BP,2)
8390 D$ =  CHR$ (4)
8400 A$ = "DOFIU"
8410  PRINT D$;"OPEN ";A$
8420  PRINT D$;"DELETE ";A$
8430  PRINT D$;"OPEN ";A$;", L20"
8450  PRINT D$;"WRITE ";A$;",R1"
8460  PRINT BP: REM  RANK OF SSM
8480  PRINT D$;"WRITE ";A$;",R2"
8490  PRINT BR%: REM  NUMBER OF ELEMENTS
8496  PRINT D$;"WRITE ";A$;",R3"
8497  PRINT CF: REM  BANDWIDTH IN NODES
8500  FOR I = 1 TO BP
8510  FOR J = 0 TO 2
8520 K = (I * 3) + J + 1
8530  PRINT D$;"WRITE ";A$;",R";K
8540  PRINT DOFIU(I,J)
8550  NEXT J
8560  NEXT I
8570  PRINT D$;"CLOSE ";A$
8580  PRINT "DOFIU SAVED IN DISK": PRINT : PRINT
8590  RETURN
8600  REM   ******************************
8610  REM  CALCULATE BANDWIDTH
8620  FOR I = 1 TO AE%
8625  FOR J = 1 TO AB%(I,3)
8630 N1 = AB%(I,4) + (AB%(I,5) * (J - 1))
8635 N2 = AB%(I,6) + (AB%(I,7) * (J - 1))
8640 B = ( ABS (N2 - N1))
8645  IF B > CF THEN CF = B
8650  NEXT J
8655  NEXT I
8656 CI = (CF + 1) * 6
8660  PRINT : PRINT "1/2 BANDWIDTH HAS BEEN CALCULATED AS (CF+1)*6= ";CI: PRINT

8665 K1 = (CI * (AD% * 6)) * 20
8670  IF K1 > 60000 THEN  PRINT "THE CAPACITY OF THE DISK STORAGE HAS BEEN
      EXCEDDED AS (BANDWIDTH*DOF)*20>60000.  CHECK TO REDUCE BANDWIDTH OR
      NUMBER OF NODES": PRINT
8673  IF K1 > 60000 THEN  STOP
8675  RETURN
8676  REM   **************************************
8700  REM  CALCULATE ADDRESS OF THE ELEMENT SSM IN DISK
8702  IF I <  = 6 THEN T1% = N1
8703  IF I > 6 THEN T1% = N2
8704  IF J <  = 6 THEN T2% = N1
8705  IF J > 6 THEN T2% = N2
8706  IF I <  = 6 THEN T3% = I
8707  IF I > 6 THEN T3% = (I - 6)
8708  IF J <  = 6 THEN T4% = J
8709  IF J > 6 THEN T4% = (J - 6)
8710 R3% = ((T1% - 1) * 6) + T3%: REM  ADDRESS OF THE DIAGONAL
8715 R4% = ((T2% - 1) * 6) + T4%: REM  ADDRESS OF TERM (ROW)
8720 R1% = R3% * CI
8725 R2% = R1% - (R3% - R4%)
8730  RETURN
8735  REM   ********************
8750  REM  SAVE BANDWIDTH CF
8753 A$ = "ID"
```

```
8755  PRINT D$;"OPEN ";A$;", L12"
8760  PRINT D$;"WRITE ";A$;",R2"
8765  PRINT CF: REM  BANDWIDTH
8770  PRINT D$;"CLOSE ";A$
8775  RETURN
8780  REM   ******************
8800  REM  STORE ESM BEAM (LOCAL)
8810 D$ =  CHR$ (4)
8820 B$ =  STR$ (BR%)
8830 A$ = "ESM" + B$
8845  PRINT D$;"OPEN ";A$
8846  PRINT D$;"DELETE ";A$
8850  PRINT D$;"OPEN ";A$;", L20"
8860  FOR I = 1 TO 6
8865 R2% = I
8870  PRINT D$;"WRITE ";A$;",R";R2%
8875  PRINT BC(I,I)
8880  NEXT I
8885  PRINT D$;"WRITE ";A$;",R7"
8890  PRINT BC(5,3)
8895  PRINT D$;"WRITE ";A$;",R8"
8897  PRINT BC(6,2)
8900  PRINT D$;"CLOSE ";A$
8905  PRINT "ESM SAVED ON DISK"
8910  RETURN
8915  REM   ***********************
```

```
5  HOME : CLEAR
10  HTAB (10): PRINT "PROGRAM ASTRA2": PRINT
20  REM  27 OCT 1980
90  DIM AAID$(1),CD(150),CE(150),BU(150),DOFIU(150,3)
100  GOSUB 1220: REM  READ DOFIU
110  GOSUB 620: REM  READ DATA
120  GOSUB 1430: REM   REARANGE DOFIU IN ORDER OF USAGE
130  PRINT "DOFIU HAVE BEEN REARANGED"
135  GOSUB 6100: REM DELETE DOFIU
140  GOSUB 410: REM   CLEAR KA,KB,KC,LA,LB
150  GOSUB 2280: REM  CALCULATE U-AA AND U-AB
155  GOSUB 6000: REM    STORE  REARANGED DOFIU
160  GOSUB 2160: REM   PRINT DOFIU AND BU
165  GOSUB 6150: REM  SAVE BU
170  GOSUB 2710: REM  ASSY OF K-AA AND K-AB
175 A$ = "SSM1X": REM  GOSUB 6310: REMCLEAR SSM
177  IF BV <  = 48 THEN  GOSUB 7000:A$ = "KA1": GOSUB 6310:A$ = "LA1": GOSUB
     6310: GOTO 250: REM  CAL LA,LB & KA,CLEAR KA,LA
180  GOSUB 3750: REM  DECOMPOSE K TO FORM L
185 A$ = "KA1": GOSUB 6310: REM CLEAR KA
190  GOSUB 4770: REM CALC L-INVERSE
195 A$ = "LA1": GOSUB 6310: REM CLEAR LA
250  PRINT "RUN ASTRA3 TO COMPLETE THE JOB"
260  END
270  REM   *********************************************
410  PRINT "CLEARING MATRICES KA,KB,KC,LA,LB
440 A$ = "KA1"
450  PRINT D$;"OPEN ";A$
460  PRINT D$;"DELETE ";A$
470 A$ = "KB1"
480  PRINT D$;"OPEN ";A$
490  PRINT D$;"DELETE ";A$
500 A$ = "KC1"
510  PRINT D$;"OPEN ";A$
520  PRINT D$;"DELETE ";A$
530 A$ = "LA1"
540  PRINT D$;"OPEN ";A$
550  PRINT D$;"DELETE ";A$
560 A$ = "LB1"
570  PRINT D$;"OPEN ";A$
580  PRINT D$;"DELETE ";A$
600  RETURN
610  REM   ************************
620  REM  SUBROUTINE READ DATA
630 D$ =  CHR$ (4): REM  CTRL D
632  PRINT D$;"OPEN ID, L12"
634  PRINT D$;"READ ID,R1"
636  INPUT AA$(0)
638  REM  PRINT D$;"READ ID,R2"
640  REM  INPUT CF
643  PRINT D$;"CLOSE ID"
650  PRINT : PRINT "READING INPUT INFORMATION"
660  PRINT D$;"OPEN ";AA$(0);", L12"
670  PRINT D$;"READ ";AA$(0);",R";1
680  INPUT AD%
690  PRINT D$;"READ ";AA$(0);",R";2
700  INPUT AE%
```

```
710  PRINT D$;"READ ";AA$(0);",R";3
720  INPUT AF%
730  PRINT D$;"READ ";AA$(0);",R";4
740  INPUT AH%
750  PRINT D$;"READ ";AA$(0);",R";5
760  INPUT AI%
765  PRINT D$;"CLOSE ";AA$(0)
766  PRINT "INPUT DATA HAS BEEN READ"
860  RETURN
870  REM   **********************************
1220  PRINT "READ DOFIU(BP,2)": REM  # ELEMENTS, RANK OF SSM
1230 D$ = CHR$ (4)
1240 A$ = "DOFIU"
1250  PRINT D$;"OPEN ";A$;", L20"
1270  PRINT D$;"READ ";A$;",R1"
1280  INPUT BP: REM   RANK OF SSM
1300  PRINT D$;"READ ";A$;",R2"
1310  INPUT BR%: REM   NUMBER OF ELEMENTS
1314  PRINT D$;"READ ";A$;",R3"
1315  INPUT CF: REM  BANDWIDTH IN NODES
1316 CG = (CF + 1) * 6: REM  BANDWIDTH IN DOF
1320  FOR I = 1 TO BP
1330  FOR J = 0 TO 2
1340 K = (I * 3) + J + 1
1350  PRINT D$;"READ ";A$;",R";K
1360  INPUT DOFIU(I,J)
1370  NEXT J
1380  NEXT I
1390  PRINT D$;"CLOSE ";A$
1400  PRINT "DOFIU READ FROM DISK"
1410  RETURN
1420  REM   ***************************************************
1430  REM  REARANGE DOFIU IN # ORDER OF USAGE
1440 K = 0: REM  RECORD/NUMBER OF DOF IN USE
1450  FOR I = 1 TO 150
1460  IF (DOFIU(I,0) = 0) GOTO 1490: REM  DOF NOT USED
1470 K = K + 1
1480 DOFIU(K,3) = I
1490  NEXT I
1500 CC = K: REM  RANK OF CONSOLIDATED MATRIX K
1510  RETURN
1520  REM   *********************************************
1530  REM  SET AND STORE NODAL SUPPRESSIONS
1540  FOR I = 1 TO AH%
1550 R% = (AG%(I,0) - 1) * 6
1560  FOR J = 1 TO 6
1570  IF AG%(I,J) = 0 GOTO 1600
1580 R1% = R% + J
1590 DOFIU(R1%,2) = 1
1600  NEXT J
1610  NEXT I
1620  RETURN
1630  REM   *****************************************************
1640  REM   STORE NODAL LOADS IN STRUCTURE LOAD MATRIX
1650  FOR I = 1 TO AI%
1660 R% = (AJ(I,0) - 1) * 6
1670  FOR J = 1 TO 6
1680  IF AG%(I,J) = 0 GOTO 1710
1690 R1% = R% + J
1700 DOFIU(R1%,1) = AJ(I,J)
```

```
1710  NEXT J
1720  NEXT I
1730  RETURN
2150  REM   *******************************************************
2160  REM  PRINT DOFIU AND BU
2170  PRINT
2180  PRINT "CHECK PRINTOUT OF DOFIU AND BU"
2190  FOR I = 1 TO BP
2200  PRINT I; SPC( 2)
2210  FOR J = 0 TO 3
2220  PRINT DOFIU(I,J); SPC( 2);
2230  NEXT J
2240  PRINT BU(I)
2250  NEXT I
2260  RETURN
2270  REM   **************************************************
2280  REM    PARTITIONING MATRICES U-A AND U-B
2290 BV = 0: REM  RANK OF UA
2300  FOR I = 1 TO CC: REM  CONSOLIDATED SSM
2310 K = DOFIU(I,3)
2320  IF DOFIU(K,2) = 1 GOTO 2350: REM  DOF IS SUPPRESSED
2330 BV = BV + 1
2340 BU(BV) = K
2350  NEXT I
2360  PRINT "THIS IS BV ";BV
2370 K1 = BV
2380  FOR I = 1 TO CC: REM  CONSOLIDATED SSM
2390 K = DOFIU(I,3)
2400  IF DOFIU(K,2) = 0 GOTO 2430: REM  DOF IS FREE
2410 K1 = K1 + 1
2420 BU(K1) = K
2430  NEXT I
2440 BW = (CC - BV): REM  RANK OF U-B
2450  PRINT : PRINT "U-A AND U-B HAVE BEEN CALCULATED"
2460  PRINT : PRINT "RANK OF CONSOLIDATED MATRIX K IS ";CC
2470  PRINT : PRINT "RANK OF PARTITIONED MATRIX K-AA IS ";BV
2480  PRINT : PRINT "RANK OF PARTITIONED MATRIX K-BB IS ";BW
2490  PRINT "MATRICES U-A AND U-B HAVE BEEN CALCULATED"
2500  RETURN
2510  REM   ****************************************
2550  NEXT J
2710  PRINT : PRINT "ASSY OF PARTITIONED SSM K-AA": PRINT
2712  REM  K-AA IS STORED AS LOWER SYMMETRIC MATRIX
2720 D$ =  CHR$ (4)
2725 N1% = 0: REM   ADDRESS OF K-AA
2728 CG = (CF + 1) * 6
2730  FOR I = 1 TO BV
2740  FOR J = 1 TO I
2770  REM  R2% IS THE ADDRESS WHERE THE VALUE IN SSM IS STORED
2780 N1% = BU(I)
2790 N2% = BU(J)
2795 BQ = 0
2800 I1% = N1% * CG
2804 I2% = N1% - N2%
2805 I3% = I2% + 1
2806  REM  PRINT "BU(I)-BU(J) IS ";I2%
2807  IF I3% > CG GOTO 2990: REM OUTSIDE BANDWIDTH
2810 R2% = I1% - I2%
2870 A$ = "SSM1X"
2880  PRINT D$;"OPEN ";A$;", L20"
```

```
2890  PRINT D$;"READ ";A$;", R";R2%
2900  INPUT BQ
2910  PRINT D$;"CLOSE ";A$
2990  REM  WRITE K-AA
2995 K1% = K1% + 1
3000 C$ = "KA1"
3090  PRINT D$;"OPEN ";C$;", L20"
3100  PRINT D$;"WRITE ";C$;", R";K1%
3110  PRINT BQ
3120  PRINT D$;"CLOSE ";C$
3130  NEXT J
3140  NEXT I
3170  PRINT "MATRIX K-AA STORED IN DISK
3180  REM   ***********************************************
3190  PRINT "ASSY OF PARTITIONED SSM K-ABT": PRINT
3200  REM  K-ABT IS STORED AS A COMPLETE MATRIX BW*BV
3210 D$ =  CHR$ (4)
3220  REM  BW IS RANK OF L-B
3230  REM  CC IS THE RANK OF SSM CONSOLIDATED
3235 K8 = BV + 1
3240  FOR I = K8 TO CC
3250  FOR J = 1 TO BV
3270  REM   CALCULATE ADDRESS OF SSM CORRESPONDING TO K-AA
3280  REM   R2% IS THE ADDRESS WHERE THE VALUE IN SSM WILL BE  STORED
3290 N1% = BU(I)
3300 N2% = BU(J)
3310  IF BU(J) > BU(I) THEN N1% = BU(J):N2% = BU(I)
3320  REM    CALCULATE THE ADDRESS OF K-ABT
3330 I1% = N1% * CG
3335 I2% = N1% - N2%
3336 I3% = I2% + 1
3337 BG = 0
3339  IF I3% > CG GOTO 3530: REM OUTSIDE BANDWIDTH
3340 R2% = I1% - I2%
3400 A$ = "SSM1X"
3410  PRINT D$;"OPEN ";A$;", L20"
3420  PRINT D$;"READ ";A$;", R";R2%
3430  INPUT BQ
3440  PRINT D$;"CLOSE ";A$
3445  REM  WRITE K-ABT
3530 C$ = "KB1"
3535 K1% = ((I - K8) * BV) + J: REM ADDRESS OF K-ABT
3620  PRINT D$;"OPEN ";C$;", L20"
3630  PRINT D$;"WRITE ";C$;", R";K1%
3640  PRINT BQ
3650  PRINT D$;"CLOSE ";C$
3670  NEXT J
3680  NEXT I
3710  PRINT "MATRIX K-ABT STORED IN DISK"
3720  RETURN
3740  REM   ***********************************************
3750  PRINT : PRINT "CALCULATION OF L-AA"
3790 D$ =  CHR$ (4)
3800  FOR I = 1 TO BV
3810  FOR J = 1 TO I
3820  REM   PRINT "CHECK  I=";I;"  J=";J
3830  IF I = J GOTO 3990
3840 X1 = 0: REM  AXL
3850 M = J - 1
3855  IF M = 0 GOTO 3912
```

```
3860  FOR K = 1 TO M
3862  REM    AXL=AXL+(LA(J,K)*LA(I,K)
3866 T3 = J:T4 = K
3868  GOSUB 6400: REM ADDRESS (J,K)
3871 A$ = "LA1"
3872  GOSUB 6450: REM LA(I,K)
3874 X2 = T8
3876 T3 = I:T4 = K: GOSUB 6400: REM ADDRESS (I,K)
3878 T2% = T5%
3880  GOSUB 6450: REM   LA(K,J)
3882 X3 = T8
3891 X1 = X1 + (X2 * X3)
3910  NEXT K
3912 T3 = I:T4 = J
3914  GOSUB 6400: REM  CAL (I,J)
3915 A$ = "KA1"
3916  GOSUB 6450: REM  CAL KA(I,J)
3917  REM  X2=KA(I,J)
3918 X2 = T6
3920 T3 = J:T4 = J
3922  GOSUB 6400: REM  CAL (J,J)
3923 A$ = "LA1"
3924  GOSUB 6450: REM  CAL LA(J,J)
3926 X3 = T8: REM
3930  REM  BQ=XL(I,J)
3940 BQ = (X2 - X1) / X3
3950  REM  CALCULATE ADDRESS (I,J)
3952 T3 = I:T4 = J: GOSUB 6400
3955 R2% = T5%
3960  GOSUB 4690: REM  WRITE LA
3970  GOTO 4140
3980  REM  I=J
3990 X1 = 0: REM  AXL
4000  IF I = 1 GOTO 4080
4010 M = I - 1
4020  FOR K = 1 TO M
4030 T3 = I:T4 = K: GOSUB 6400: REM  ADDRESS (I,K)
4032 A$ = "LA1"
4034  GOSUB 6450: REM  READ LA(I,K)
4036 X2 = T8
4051 X1 = X1 + (X2 ↑ 2)
4060  NEXT K
4080 T3 = I:T4 = I: GOSUB 6400: REM  ADDRESS (I,I)
4084 A$ = "KA1"
4086  GOSUB 6450: REM  READ KA(I,I)
4088 X2 = T8
4090 BQ = (X2 - X1) ↑ .5
4094 R2% = T5%
4096  GOSUB 4690: REM  WRITE LA(I,I)
4140  NEXT J
4150  NEXT I
4180  PRINT "DECOMPOSITION OF K INTO L IS COMPLETE"
4190  RETURN
4680  REM   *********
4690  REM  WRITE MATRIX LA
4695 A$ = "LA1"
4700  PRINT D$;"OPEN ";A$;", L20"
4710  PRINT D$;"WRITE ";A$;", R";R2%
4720  PRINT BQ
4730  PRINT D$;"CLOSE ";A$
```

```
4740  RETURN
4750  REM   ****************************************
4770  PRINT : PRINT "CALCULATION OF L-AA INVERSE"
4790 D$ =  CHR$ (4)
4800  FOR I = 1 TO BV
4810  REM PRINT "CHECK  I=";I;"  J=";I
4820 T3 = I:T4 = I: GOSUB 6400: REM ADDRESS (I,I)
4825 A$ = "LA1": GOSUB 6450: REM READ LA(I,I)
4827 X1 = T6
4830 BG = 1 / X1
4850  GOSUB 5520: REM  WRITE LB
4860  IF I = 1 GOTO 5020
4870 M = I - 1
4880  FOR J = 1 TO M
4890  REM  PRINT "CHECK  I=";I;"  J=";J
4900 X1 = 0: REM   AXL
4910  FOR K = J TO M
4920 T3 = I:T4 = K: GOSUB 6400: REM ADDRESS (I,K)
4925 A$ = "LA1": GOSUB 6450: REM  READ LA(I,K)
4927 X2 = T6
4930 T3 = K:T4 = J: GOSUB 6400: REM ADDRESS (K,J)
4935 A$ = "LB1": GOSUB 6450: REM  READ LB(K,J)
4940 X3 = T6
4941 X1 = X1 + (X2 * X3)
4960  NEXT K
4970 T3 = I:T4 = I: GOSUB 6400: REM ADDRESS (I,I)
4975 A$ = "LA1": GOSUB 6450: REM  READ LA(I,I)
4980 X2 = T6
4985 BG =  - X1 / X2
4990 T3 = I:T4 = J: GOSUB 6400: REM ADDRESS (I,J)
5000  GOSUB 5520: REM  WRITE LB
5010  NEXT J
5020  NEXT I
5050  PRINT "CALCULATION OF L INVERTED IS COMPLETE"
5060  RETURN
5510  REM   *********
5520  REM   WRITE MATRIX LB
5525 A$ = "LB1"
5530  PRINT D$;"OPEN ";A$;", L20"
5540  PRINT D$;"WRITE ";A$;", R";T5%
5550  PRINT BG
5560  PRINT D$;"CLOSE ";A$
5570  RETURN
5580  REM   ****************************************
6000  REM  WRITE DOFIU(I,3)
6005 A$ = "DOFIU"
6007  PRINT D$;"OPEN ";A$;", L10"
6008  PRINT D$;"WRITE ";A$;",R1"
6009  PRINT BF: REM    RANK OF SSM
6010  PRINT D$;"WRITE ";A$;",R2"
6011  PRINT BR%: REM  NUMBER OF ELEMENTS
6012  PRINT D$;"WRITE ";A$;",R3"
6013  PRINT CC: REM  RANK OF CONSOLIDATED MATRIX
6015  PRINT D$;"WRITE ";A$;",R0"
6016  PRINT BV: REM   RANK OF UA
6018  FOR I = 1 TO BF
6020  FOR J = 0 TO 3
6030 K = (I * 4) + J
6040  PRINT D$;"WRITE ";A$;",R";K
6050  PRINT DOFIU(I,J)
```

```
6060  NEXT J
6070  NEXT I
6080  PRINT D$;"CLOSE ";A$
6090  PRINT "DOFIU SAVED IN DISK"
6095  RETURN
6097  REM   ******************
6100  REM  DELETE DOFIU
6105  PRINT D$;"OPEN DOFIU"
6107  PRINT D$;"DELETE DOFIU"
6108  RETURN
6150  REM  WRITE BU
6160 A$ = "BU"
6170  PRINT D$;"OPEN ";A$;", L4"
6180  FOR I = 1 TO BF
6190  PRINT D$;"WRITE ";A$;",R";I
6200  PRINT BU(I)
6210  NEXT I
6220  PRINT D$;"CLOSE ";A$
6230  PRINT "BU SAVED IN DISK"
6240  RETURN
6250  REM  ******************
6310  REM    CLEAR SSM,LA
6340  PRINT D$;"OPEN ";A$
6350  PRINT D$;"DELETE ";A$
6370  RETURN
6399  REM   **************
6400  REM   CALCULATE ADDRESS OF KA,KB,LA,LB
6410 T5% = 0
6415 T6 = T3 - 1
6417  IF T6 = 0 GOTO 6435
6420  FOR K8 = 1 TO T6
6425 T5% = T5% + K8
6430  NEXT K8
6435 T5% = T5% + T4
6440  RETURN
6445  REM   *****************
6450  REM   READ MATRICES (KA,LA,LB
6452  PRINT D$;"OPEN ";A$;", L20"
6455  PRINT D$;"READ ";A$;", R";T5%
6460  INPUT T8
6465  PRINT D$;"CLOSE ";A$
6470  RETURN
6475  REM   ******************
7000  PRINT "CALCULATION OF LA,LB WHEN BV<=48 USING RAM": PRINT
7002  IF BV > 48 THEN  STOP
7003 D$ =  CHR$ (4)
7005  REM  CALCULATE SIZE OF LA,LB
7010 CI = 0
7015  FOR I = 1 TO BV
7020 CI = CI + I
7025  NEXT I
7026  REM  ALLOCATE ARRAYS
7027  DIM KA1(CI),LA1(CI),LB1(CI)
7029  REM    ******************
7030  REM   READ KA
7031 K1 = 0
7035 A$ = "KA1"
7037 D$ =  CHR$ (4)
7040  PRINT D$;"OPEN ";A$;", L20"
7045  FOR I = 1 TO BV
```

```
7050  FOR J = 1 TO I
7055 K1 = K1 + 1
7060  PRINT D$;"READ ";A$;", R";K1
7065  INPUT KA1(K1)
7070  NEXT J
7075  NEXT I
7080  PRINT D$;"CLOSE ";A$
7082  REM   *****************
7085  REM  CALCULATE LA
7090  FOR I = 1 TO BV
7095  FOR J = 1 TO I
7100  REM  PRINT "CHECK  I=";I;"  J=";J
7105  IF I = J GOTO 7190
7110 X1 = 0: REM   AXL
7115 M = J - 1
7120  IF M = 0 GOTO 7150
7125  FOR K = 1 TO M
7126  REM   AXL=AXL+(LA1(J,K)*LA1(K,J))
7127 T3 = J:T4 = K: GOSUB 6400: REM ADDRESS (J,K)
7130 X2 = LA1(T5%)
7133 T3 = I:T4 = K: GOSUB 6400: REM ADDRESS (I,K)
7135 X3 = LA1(T5%)
7140 X1 = X1 + (X2 * X3)
7145  NEXT K
7150 T3 = J:T4 = J: GOSUB 6400: REM  CAL (J,J)
7153 X3 = LA1(T5%)
7155 T3 = I:T4 = J: GOSUB 6400: REM ADDRESS (I,K)
7160 X2 = KA1(T5%)
7170 LA1(T5%) = (X2 - X1) / X3
7175  GOTO 7255
7180  REM   I=J
7190 X1 = 0: REM   AXL
7200  IF I = 1 GOTO 7232
7205 M = I - 1
7210  FOR K = 1 TO M
7215 T3 = I:T4 = K: GOSUB 6400: REM ADDRESS (I,K)
7220 X2 = LA1(T5%)
7225 X1 = X1 + (X2 ↑ 2)
7230  NEXT K
7232 T3 = I:T4 = I: GOSUB 6400: REM ADDRESS (I,I)
7235 X2 = KA1(T5%)
7250 LA1(T5%) = (X2 - X1) ↑ .5
7255  NEXT J
7260  NEXT I
7263  GOSUB 7520: REM  STORE LA1
7265  PRINT "DECOMPOSITION OF K INTO L IS COMPLETE"
7299  REM   ****************************************
7300  PRINT "CALCULATE LB USING RAM": PRINT
7310  FOR I = 1 TO BV
7315  REM  PRINT "CHECK  I=";I;"  J=";I
7320 T3 = I:T4 = I: GOSUB 6400: REM ADDRESS (I,I)
7335 LB1(T5%) = 1 / LA1(T5%)
7340  IF I = 1 GOTO 7415
7345 M = I - 1
7350  FOR J = 1 TO M
7355  REM   PRINT "CHECK  I=";I;"  J=";J
7360 X1 = 0: REM    AXL
7365  FOR K = J TO M
7370 T3 = I:T4 = K: GOSUB 6400: REM ADDRESS (I,K)
7375 X2 = LA1(T5%)
```

```
7380 T3 = K:T4 = J: GOSUB 6400: REM ADDRESS (K,J)
7385 X3 = LB1(T5%)
7390 X1 = X1 + (X2 * X3)
7395  NEXT K
7400 T3 = I:T4 = I: GOSUB 6400: REM ADDRESS (I,I)
7401 X2 = LA1(T5%)
7403 T3 = I:T4 = J: GOSUB 6400: REM  ADDRESS (I,J)
7405 LB1(T5%) =  - X1 / X2
7410  NEXT J
7415  NEXT I
7420  REM  STORE LB IN DISK
7422 A$ = "LB1"
7424  PRINT D$;"OPEN ";A$;", L20"
7426  FOR I = 1 TO CI
7428  PRINT D$;"WRITE ";A$;", R";I
7430  PRINT LB1(I)
7432  NEXT I
7434  PRINT D$;"CLOSE ";A$
7440  PRINT "CALCULATION OF L INVERTED IS COMPLETE"
7445  RETURN
7500  REM  ******************
7520  REM   STORE LA IN DISK
7522 A$ = "LA1"
7524  PRINT D$;"OPEN ";A$;", L20"
7526  FOR I = 1 TO CI
7528  PRINT D$;"WRITE ";A$;", R";I
7530  PRINT LA1(I)
7532  NEXT I
7534  PRINT D$;"CLOSE ";A$
7540  PRINT "LA1 STORED IN DISK"
7545  RETURN
```

```
10  REM   PROGRAM ASTRA3
20  REM   25 NOVEMBER 1980
30  HOME : CLEAR
35  DIM AAID$(5)
40  GOSUB 620: REM   READ DATA
50  DIM CD(300),CE(300),BU(300),DOFIU(300,3)
60  GOSUB 1220: REM   READ DOFIU
70  GOSUB 7400: REM   READ BU
115  REM_ PRINT BF,BR%,CC,BV
120  IF BV < = 48 THEN  GOSUB 8000: GOTO 203: REM  CALC KC USING RAM
200  GOSUB 5590: REM CALCULATE KA INVERSE
203  REM  GOSUB 3100: REM  CLEAR LB
210  GOSUB 6160: REM    CALCULATE NODAL DEFLECTIONS
220  GOSUB 6700: REM   CALCULATE REACTIONS
230  GOSUB 7220: REM  PRINT REACTIONS
240  GOSUB 9000: REM  STORE NODAL DEFLECTIONS
250  PRINT "RUN ASTRA4 TO COMPLETE JOB"
260  END
270  REM   ***********************************************
620  REM  SUBROUTINE READ DATA
630 D$ =  CHR$ (4): REM  CTRL D
640  PRINT D$;"OPEN ID, L12"
641  PRINT D$;"READ ID,R1"
642  INPUT AA$(0)
643  PRINT D$;"CLOSE ID"
650  PRINT : PRINT "READING INPUT INFORMATION
660  PRINT D$;"OPEN ";AA$(0);", L12"
670  PRINT D$;"READ ";AA$(0);",R";1
680  INPUT AD%
690  PRINT D$;"READ ";AA$(0);",R";2
700  INPUT AE%
710  PRINT D$;"READ ";AA$(0);",R";3
720  INPUT AF%
730  PRINT D$;"READ ";AA$(0);",R";4
740  INPUT AH%
750  PRINT D$;"READ ";AA$(0);",R";5
760  INPUT AI%
800  PRINT D$;"CLOSE ";AA$(0)
860  RETURN
870  REM   ************************************
1120  REM     READ DOFIU
1130 D$ =  CHR$ (4)
1140 A$ = "DOFIU"
1150  PRINT D$;"OPEN ";A$;", L10"
1170  PRINT D$;"READ ";A$;",R1"
1180  INPUT BP: REM    RANK OF SSM
1190  PRINT D$;"READ ";A$;",R2"
1192  INPUT BR%: REM    NUMBER OF ELEMENTS
1193  PRINT D$;"READ ";A$;",R3"
1194  INPUT CC: REM   RANK OF CONSOLIDATED MATRIX
1195  PRINT D$;"READ ";A$;",R0"
1196  INPUT BV: REM   RANK OF JA
1197  PRINT D$;"CLOSE ";A$
1198  RETURN
1199  REM   ********************************************
1220  REM     READ DOFIU(BF,3):REM # ELEMENTS, RANK OF SSM
1230  GOSUB 1120
```

```
1240  PRINT D$;"OPEN ";A$;", L10"
1320  FOR I = 1 TO BF
1330  FOR J = 0 TO 3
1340 K = (I * 4) + J
1350  PRINT D$;"READ ";A$;",R";K
1360  INPUT DOFIU(I,J)
1370  NEXT J
1380  NEXT I
1390  PRINT D$;"CLOSE ";A$
1400  PRINT "DOFIU READ IN DISK"
1410  RETURN
1420  REM  **************************************************
2060  PRINT "ERROR-FIND ADDRESS OF KA": GOTO 2080
2070  PRINT "ERROR-FIND ADDRESS OF KB": GOTO 2080
2080  STOP
2090  REM   **************************************************
2160  REM  PRINT DOFIU AND BU
2170  PRINT
2180  PRINT "CHECK PRINTOUT OF DOFIU AND BU"
2190  FOR I = 1 TO BF
2200  PRINT I; SPC( 2)
2210  FOR J = 0 TO 3
2220  PRINT DOFIU(I,J); SPC( 2);
2230  NEXT J
2240  PRINT BU(I)
2250  NEXT I
2260  RETURN
2270  REM   **************************************************
3000  REM  CLEAR L-AA
3010  FOR I = 1 TO 3
3015 B$ =  STR$ (I)
3020 A$ = "LA" + B$
3025  PRINT D$;"OPEN ";A$
3030  PRINT D$;"DELETE ";A$
3035  NEXT I
3040  RETURN
3090  REM    ********************
3100  REM   CLEAR L-AA INV.
3120 A$ = "LB1"
3125  PRINT D$;"OPEN ";A$
3130  PRINT D$;"DELETE ";A$
3140  RETURN
3145  REM    ********************
4000  REM  FIND ADDRESS OF LB,KC
4010 T5% = 0
4015 T6 = T3% - 1
4017  IF T6 = 0 GOTO 4035
4020  FOR K8 = 1 TO T6
4025 T5% = T5% + K8
4030  NEXT K8
4035 T5% = T5% + T4%
4040  RETURN
4050  REM   ***************
4100  REM  READ MATRIX LB
4105 D$ =  CHR$ (4)
4110  PRINT D$;"OPEN ";A$;", L20"
4120  PRINT D$;"READ ";A$;", R";T5%
4130  INPUT T8
4140  PRINT D$;"CLOSE ";A$
4150  RETURN
```

```
5580  REM   ********************************************
5590  PRINT "CALCULATION OF K-AA INVERSE"
5605  REM KC(K1,K2)=KC(K1,K2)+(LBT(K1,K3)*LB(K3,K2))
5607 R2% = 0
5610  FOR K1 = 1 TO BV
5620  FOR K2 = 1 TO K1
5630  REM  PRINT "CALCULATE K-A INVERSE ( ";K1;",";K2;" )"
5640 BG = 0
5650  FOR K3 = 1 TO BV
5652  IF K1 > K3 GOTO 5760: REM BECAUSE OF 0 TERMS IN LB
5654  IF K2 > K3 GOTO 5760: REM BECAUSE OF 0 TERMS IN LB
5660  REM_  FIND ADDRESS OF LB TRANSPOSE
5665 T3% = K3:T4% = K1: GOSUB 4000: REM  ADDRESS LBT=LB(K3,K1)
5670 A$ = "LB1": GOSUB 4100: REM   READ LBT
5675 X2 = T6
5710  REM  FIND ADRESS OF LB
5715  IF K2 = K1 THEN BG = BG + (X2 ↑ 2): GOTO 5760
5740 T3% = K3:T4% = K2: GOSUB 4000: REM  ADDRESS LB(K3,K2)
5745 A$ = "LB1": GOSUB 4100: REM READ LB
5747 X3 = T6
5750 BG = BG + (X2 * X3)
5760  NEXT K3
5800 R2% = R2% + 1: REM ADDRESS KC
5810  GOSUB 6055: REM  WRITE KC
5820  NEXT K2
5840  NEXT K1
5845  PRINT "K-A INVERSE HAS BEEN CALCULATED": PRINT
5860  RETURN
5870  REM   ***********
6055  REM  WRITE KC
6056 A$ = "KC1"
6060  PRINT D$;"OPEN ";A$;", L20"
6070  PRINT D$;"WRITE ";A$;", R";R2%
6080  PRINT BG
6090  PRINT D$;"CLOSE ";A$
6100  RETURN
6170  REM   ***********************************************************
6180  PRINT "CALCULATE NODAL DEFLECTIONS"
6190  REM  MARK NODES SUPPRESSED
6200  FOR I = 1 TO BF
6210  IF DOFIU(I,2) = 1 THEN DOFIU(I,2) = 9999
6220 DOFIU(I,3) = 0
6230  NEXT I
6240  REM  STORE LOADS IN MATRIX CE
6250  FOR I = 1 TO BV
6260 K = BU(I)
6270 CE(I) = DOFIU(K,1)
6280  NEXT I
6290  REM  CALCULATE DEFLECTIONS
6300  FOR I = 1 TO BV
6310 BG = 0
6340  FOR J = 1 TO BV
6350  GOSUB 6470: REM  READ KC
6380  NEXT J
6390  FOR L = 1 TO BV
6400 BG = BG + (CD(L) * CE(L))
6410  NEXT L
6415 K = BU(I)
6420 DOFIU(K,2) = BG
6430  NEXT I
```

```
6450  RETURN
6460  REM   ***********
6470  REM  READ KC
6472  IF I >  = J THEN T3% = I:T4% = J
6473  IF I < J THEN T3% = J:T4% = I
6474  GOSUB 4000: REM  FIND ADDRESS
6540 A$ = "KC1"
6550  PRINT D$;"OPEN ";A$;", L20"
6560  PRINT D$;"READ ";A$;", R";T5%
6570  INPUT CD(J)
6580  PRINT D$;"CLOSE ";A$
6610  RETURN
6690  REM   *********************************************
6700  PRINT "CALCULATE REACTIONS"
6710  REM   STORE DEFLECTIONS IN MATRIX CE
6720  FOR I = 1 TO BV
6730 K = BU(I)
6740 CE(I) = DOFIU(K,2)
6750  IF CE(I) = 9999 THEN CE(I) = 0
6760  NEXT I
6770  PRINT "DEFLECTIONS HAVE BEEN STORED"
6780  REM    CALCULATE REACTIONS
6785 K8 = BV + 1
6790  FOR I = K8 TO CC
6800 BG = 0
6810 K = I - BV
6830  FOR J = 1 TO BV
6840  GOSUB 6970: REM  READ KB
6870  NEXT J
6880  FOR L = 1 TO BV
6890 BG = BG + (CD(L) * CE(L))
6900  NEXT L
6910 K = BU(I)
6920 DOFIU(K,3) = BG
6930  NEXT I
6950  RETURN
6960  REM    ***********
6970  REM  READ KB
6980 R2% = ((K - 1) * BV) + J
7040 A$ = "KB1"
7060  PRINT D$;"OPEN ";A$;", L20"
7070  PRINT D$;"READ ";A$;", R";R2%
7080  INPUT CD(J)
7090  PRINT D$;"CLOSE ";A$
7110  RETURN
7210  REM    *******************************************************
    *****
7220  PRINT "PRINT NODAL LOAD, DEFLECTIONS AND REACTIONS"
7230 K = 0
7240  PRINT : PRINT "NODE #"; SPC( 10);"DOF"; SPC( 10);"NODE LOAD"; SPC( 1
    0);"DEFLECTIONS"; SPC( 10);"REACTIONS": PRINT
7250  FOR I = 1 TO BP STEP 6
7260 K = K + 1: REM  NODE NUMBER
7270  FOR J = 1 TO 6
7280 J1 = (K - 1) * 6 + J
7290  PRINT K;
7300  HTAB (16): PRINT J;
7310  HTAB (30): PRINT DOFIU(J1,1);
7320  IF DOFIU(J1,2) = 9999 THEN  HTAB (40): PRINT  SPC( 5);"****";
7330  IF DOFIU(J1,2) <  > 9999 THEN  HTAB (40): PRINT  SPC( 5);DOFIU(J1,2)
```

```
     ;
7340  HTAB (30): PRINT  SPC( 20);DOFIU(J1,3)
7350  NEXT J
7360  PRINT
7370  NEXT I
7380  RETURN
7399  REM   ******************************
7400  REM  READ BU
7405 A$ = "BU"
7410 D$ = "": REM  CTRL D
7415  PRINT D$;"OPEN ";A$;", L4"
7420  FOR I = 1 TO BF
7425  PRINT D$;"READ ";A$;",R";I
7430  INPUT BU(I)
7435  NEXT I
7440  PRINT D$;"CLOSE ";A$
7445  PRINT "BU READ FROM DISK"
7450  RETURN
7455  REM   ******************************
8000  PRINT : PRINT "CALCULATE KC USING RAM"
8005 CI = 0: REM  SIZE OF ARRAY LB
8010  FOR I = 1 TO BV
8015 CI = CI + I
8020  NEXT I
8029  DIM KC1(CI)
8030  DIM LB1(CI)
8031 D$ =  CHR$ (4)
8035  REM  READ LB
8037 A$ = "LB1"
8040  PRINT D$;"OPEN ";A$;", L20"
8045  FOR I = 1 TO CI
8050  PRINT D$;"READ ";A$;",R";I
8055  INPUT LB1(I)
8060  NEXT I
8065  PRINT D$;"CLOSE ";A$
8070  REM   CALCULATION OF KC USING RAM
8075  REM   KC(K1,K2)=KC(K1,K2)+(LBT(K1,K3)*LB(K3,K2))
8080 R2% = 0
8081  PRINT "TEST BV= ";BV
8085  FOR K1 = 1 TO BV
8090  FOR K2 = 1 TO K1
8095  REM  PRINT "CALCULATE K-A INVERSE (";K1;",";K2;")"
8100 BG = 0
8105  FOR K3 = 1 TO BV
8110  IF K1 > K3 GOTO 8160: REM BECAUSE OF 0 TERMS IN LB
8115  IF K2 > K3 GOTO 8160: REM BECAUSE OF 0 TERMS IN LB
8120  REM    FIND ADDRESS OF LB TRANSPOSE
8125 T3% = K3:T4% = K1: GOSUB 4000: REM   ADDRESS LBT=LB(K3,K1)
8130 X2 = LB1(T5%)
8135  REM   FIND ADRESS OF LB
8140  IF K2 = K1 THEN BG = BG + (X2 ^ 2): GOTO 8160
8145  IF K2 = K1 GOTO 5760
8150 T3% = K3:T4% = K2: GOSUB 4000: REM   ADDRESS LB(K3,K2)
8155 X3 = LB1(T5%)
8157 BG = BG + (X2 * X3)
8160  NEXT K3
8165 R2% = R2% + 1: REM  ADDRESS KC
8170 KC1(R2%) = BG
8175  NEXT K2
8180  NEXT K1
```

```
8200  PRINT "STORE KC1 IN DISK": PRINT
8205 D$ =  CHR$ (4)
8210  PRINT D$;"OPEN KC1"
8215  PRINT D$;"DELETE KC1"
8220  PRINT D$;"OPEN KC1, L20"
8225  FOR I = 1 TO R2%
8230  PRINT D$;"WRITE KC1, R";I
8235  PRINT KC1(I)
8240  NEXT I
8250  PRINT D$;"CLOSE KC1"
8255  PRINT : PRINT "K-A INVERSE HAS BEEN CALCULATED" : PRINT
8260  RETURN
8990  REM   ****************************************
9000  PRINT "STORE NODAL DEFLECTIONS"
9001 D$ =  CHR$ (4)
9003 BP = AD% * 6: REM  # OF DOF
9004  PRINT D$;"OPEN ND"
9005  PRINT D$;"DELETE ND"
9006  PRINT D$;"OPEN ND, L20"
9010  FOR I = 1 TO BP
9015  PRINT D$;"WRITE ND, R";I
9020  PRINT DOFIU(I,2)
9030  NEXT I
9040  PRINT D$;"CLOSE ND"
9050  RETURN
```

```
10  HOME : CLEAR : PRINT "PROGRAM ASTRA4": PRINT
20  REM   25 NOVEMBER 1980
70  PRINT
90  GOSUB 280: REM   ALLOCATE ARRAYS
110  GOSUB 620: REM  READ DATA
230  GOSUB 9000: REM  READ NODAL DEFLECTIONS
240  GOSUB 7400: REM   CALCULATION OF MEMBER FORCES
250  HOME : PRINT "END OF JOB"
260  END
270  REM  ****************************************************************
     ****************************************************************
     ******************
280  REM   ALLOCATION OF ARRAYS I
290  DIM AAID$(5),ABEGEN%(25,9)
300  DIM ND(300): REM   NODAL  DEFLECTIONS
330  DIM BB(12,12): REM  TRANS. MATRIX
340  DIM BC(12,12): REM   STIFF. MATRIX (ELEMENT-LOCAL)
350  DIM BF(12): REM  ELEMENT FORCES AT THE NODES
360  DIM BH(12): REM   HOLD NODAL DEFLECTIONS
370  DIM BT(12): REM   BT=BB*BH USED TO CALCULATE ELEMENT FORCES
380  RETURN
390  REM   *********************************************
620  REM  SUBROUTINE READ DATA
630 D$ =  CHR$ (4): REM  CTRL D
640  GOSUB 880
650  PRINT : PRINT "READING INPUT INFORMATION"
660  PRINT D$;"OPEN ";AA$(0);", L12"
670  PRINT D$;"READ ";AA$(0);",R";1
680  INPUT AD%
690  PRINT D$;"READ ";AA$(0);",R";2
700  INPUT AE%
710  PRINT D$;"READ ";AA$(0);",R";3
720  INPUT AF%
730  PRINT D$;"READ ";AA$(0);",R";4
740  INPUT AH%
750  PRINT D$;"READ ";AA$(0);",R";5
760  INPUT AI%
770  REM  READ ELEMENT CARDS
780  FOR I = 1 TO AE%
790 J = ((I - 1) * 9) + 10
800  FOR K = 1 TO 9
810  PRINT D$;"READ ";AA$(0);",R";(J + K)
820  INPUT AB%(I,K)
830  NEXT K
840  NEXT I
850  PRINT D$;"CLOSE";AA$(0)
860  RETURN
870  REM   **********************************
880  REM  READ HEADING
890 D$ =  CHR$ (4)
900  PRINT D$;"OPEN ID, L12"
920  PRINT D$;"READ ID, R1"
930  INPUT AA$(0)
950  PRINT D$;"CLOSE ID"
960  RETURN
980  REM   ********************************************
1000  REM  READ ESM BEAM
1010 B$ =  STR$ (BR%)
```

```
1015 A$ = "ESM" + B$
1020  PRINT D$;"OPEN ";A$;", L20"
1030  FOR I = 1 TO 6
1035 R2% = I
1040  PRINT D$;"READ ";A$;", R";R2%
1045  INPUT BC(I,I)
1050  NEXT I
1060  PRINT D$;"READ ";A$;", R7"
1065  INPUT BC(5,3)
1070  PRINT D$;"READ ";A$;", R8"
1075  INPUT BC(6,2)
1077  PRINT D$;"CLOSE ";A$
1080  FOR I = 1 TO 6
1085 J = I + 6
1090 BC(J,J) = BC(I,I)
1095 BC(J,I) =  - BC(I,I)
1100  NEXT I
1105 BC(11,5) = BC(5,5) / 2
1110 BC(12,6) = BC(6,6) / 2
1115 BC(9,5) =  - BC(5,3)
1120 BC(8,6) =  - BC(6,2)
1125 BC(11,3) = BC(5,3)
1130 BC(12,2) = BC(6,2)
1135 BC(11,9) =  - BC(5,3)
1140 BC(12,8) =  - BC(6,2)
1145  FOR I = 2 TO 12
1150 K = I - 1
1155  FOR J = 1 TO K
1160 BC(J,I) = BC(I,J)
1165  NEXT J
1170  NEXT I
1175  RETURN
1180  REM   ****************************
2000  REM  CLEAR MATRICES BB,BC
2010  FOR I = 1 TO 12
2012 BF(I) = 0:BH(I) = 0:BT(I) = 0
2015  FOR J = 1 TO 12
2020 BB(I,J) = 0:BC(I,J) = 0
2025  NEXT J
2030  NEXT I
2035  RETURN
7390  REM   ***********************************************************
7400  PRINT "CALCULATION OF ELEMENT FORCES" : PRINT
7410 BR% = 0: REM  ELEMENT COUNTER
7415  PRINT "THE MEMBER FORCES WILL BE PRINTED OUT IN FREE FORMAT IN THE O
     RDER SHOWN BELOW.  THIS IS BECAUSE OF FORMAT LIMITATION WITHIN THE BA
     SIC PROGRAMMING LANGUAGE" : PRINT : PRINT
7420  PRINT "ELEMT #"; SPC( 2);"NODE #"; SPC( 5);"PX"; SPC( 10);"PY"; SPC(
     10);"PZ"; SPC( 10);
7430  PRINT "MX"; SPC( 10);"MY"; SPC( 10);"MZ" : PRINT
7435  REM   ******************************
7440  FOR I8 = 1 TO AE%: REM  # OF ELEMT CARDS
7445 NT% = AB%(I8,1)
7450  FOR I9 = 1 TO AB%(I8,3): REM  ELEMT/CARD
7460 BR% = BR% + 1
7470 N1 = AB%(I8,4) + (AB%(I8,5) * (I9 - 1))
7480 N2 = AB%(I8,6) + (AB%(I8,7) * (I9 - 1))
7485  IF N1 > N2 THEN N3 = N1:N1 = N2:N2 = N3
7487  GOSUB 2000: REM  CLEAR MATRICES BB,BC
7490  IF AB%(I8,1) = 1 THEN  GOSUB 8500: REM READ ESM ROI
```

```
7495  IF AB%(I8,1) = 2 THEN  GOSUB 1000: REM   READ ESM BEAM
7500  GOSUB 7600: REM  READ ETM
7510  GOSUB 7940: REM  READ NODAL DEFLECTIONS
7520  GOSUB 8070: REM  CALCULATE FORCES
7530  GOSUB 8270: REM    PRINT ELEMENT LOADS
7540  NEXT I9
7560  NEXT I8
7580  RETURN
7590  REM   ********************************************************
7600  REM  LOAD ETM
7605  REM  ONLY THE TOP 3X3 ETM IS STORED
7610 B$ = STR$ (BR%)
7615 R2% = 0
7620 A$ = "ETM" + B$
7622  PRINT D$;"OPEN ";A$;", L20"
7625  FOR I = 1 TO 3
7630  FOR J = 1 TO 3
7655 R2% = R2% + 1
7700  PRINT D$;"READ ";A$;", R";R2%
7710  INPUT BB(I,J)
7715  NEXT J
7720  NEXT I
7725  PRINT D$;"CLOSE ";A$
7730  FOR I = 1 TO 3
7735  FOR J = 1 TO 3
7739 I1 = I + 3:I2 = I + 6:I3 = I + 9
7740 J1 = J + 3:J2 = J + 6:J3 = J + 9
7745 BB(I2,J2) = BB(I,J)
7750  IF AB%(I8,1) = 2 THEN BB(I1,J1) = BB(I,J):BB(I3,J3) = BB(I,J)
7755  NEXT J
7760  NEXT I
7839  RETURN
7840  REM   ******************************************
7940  REM  READ NODAL DEFLECTIONS
7950  FOR I = 1 TO 6
7960 K = ((N1 - 1) * 6) + I
7970  IF ND(K) = 9999 THEN BH(I) = 0: GOTO 7990
7980 BH(I) = ND(K)
7990  NEXT I
8000  FOR I = 7 TO 12
8010 K = ((N2 - 1) * 6) + I - 6
8020  IF ND(K) = 9999 THEN BH(I) = 0: GOTO 8040
8030 BH(I) = ND(K)
8040  NEXT I
8050  RETURN
8060  REM   ********************************************
8070  REM  CALCULATION OF LOADS
8080  REM   BT=BB*BH
8140  FOR K1 = 1 TO 12
8150  FOR K3 = 1 TO 12
8160 BT(K1) = BT(K1) + (BB(K1,K3) * BH(K3))
8170  NEXT K3
8180  NEXT K1
8190  REM  BF=BC*BT
8200  FOR K1 = 1 TO 12
8210  FOR K3 = 1 TO 12
8220 BF(K1) = BF(K1) + (BC(K1,K3) * BT(K3))
8230  NEXT K3
8240  NEXT K1
8250  RETURN
```

```
8260  REM   ****************************************
8270  REM  PRINT ELEMENT FORCES
8280  PRINT BR%; SPC( 2);
8290  FOR I = 1 TO 12 STEP 6
8300  IF I < = 6 THEN  PRINT N1;: GOTO 8320
8310  PRINT  SPC( 4);N2;
8320  PRINT  SPC( 2);BF(I);
8330  PRINT  SPC( 2);BF(I + 1);
8340  PRINT  SPC( 2);BF(I + 2);
8350  PRINT  SPC( 2);BF(I + 3);
8360  PRINT  SPC( 2);BF(I + 4);
8370  PRINT  SPC( 2);BF(I + 5)
8390  NEXT I
8395  PRINT
8400  RETURN
8410  REM   **************************************************
8500  REM  LOAD ESM FOR ROD
8510 B$ =  STR$ (BR%)
8515 A$ = "ESM" + B$
8520  PRINT D$;"OPEN ";A$;", L20"
8530  PRINT D$;"READ ";A$;", R1"
8540  INPUT BC(1,1)
8550  PRINT D$;"CLOSE ";A$
8560 BC(7,7) = BC(1,1)
8570 BC(7,1) =  - BC(1,1)
8580 BC(1,7) =  - BC(1,1)
8590  RETURN
8595  REM   *********************************************
9000  PRINT "READ NODAL DEFLECTIONS"
9005 BF = AD% * 6: REM  # OF DOF
9010 D$ =  CHR$ (4)
9015  PRINT D$;"OPEN ND, L20"
9020  FOR I = 1 TO BF
9030  PRINT D$;"READ ND, R";I
9040  INPUT ND(I)
9050  NEXT I
9060  PRINT D$;"CLOSE ND"
9070  RETURN
9080  REM   ***************************************
```

APPENDIX E

EXAMPLE PROBLEMS

The six example problems solved using ASTRA and chosen to verify its accuracy are illustrated in Figures 13 through 18. Problems 1 and 2 and 3 represent a plane truss, shown in Figure 13, with problem 1 illustrating the "hand" solution and 2 the corresponding computer solution. Problem 3 is the IASTRA run to prepare the input data for ASTRA. The structures analyzed in example problems 4 through 7 include a plane truss, a space truss, a plane frame and a space frame and are illustrated in Figures 14 through 18 respectively. Solutions to these problems were calculated using ASTRA and verified with solutions obtained previously using other finite element computer programs. The figures show the structures to be analyzed, including structure dimensions, node and element numbering sequence, applied loads and support restraints. The computer outputs are listed in the following pages and includes the input data, calculated results, as well as program comments to let the user known how far along the computer solution has progressed.

Example Problem 1: Hand Solution

In this example problem, the truss shown in Figure 13a will be analysed using hand calculations and the matrix displacement method. In order to construct the idealized structure, the truss is divided into the three elements shown in Figure 13b joined at three nodes.

To obtain the structure stiffness matrix $[K_S]$ we start with the element stiffness and assume two degrees of freedom (DOF) at each node: two translations u and v in the X and Y directions. The stiffness matrix for the elements can be found in Table III.

$$[K] = \left(\frac{AE}{L}\right) \begin{bmatrix} 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

The force matrix [Q] and displacement matrix [S] are constructed from Equation (3.24)

$$[\bar{Q}] = [k] [\bar{S}]$$

The element transformation matrices are assembled from Equations (4.42) and (4.43) by deleting the degrees of freedom that are not needed.

For element 1, the direction cosines Cx and Cy and element transformation and stiffness matrices are calculated as follows:

$$L = \sqrt{(X_j - X_i)^2 + (Y_j - Y_i)^2} = 100 = 10$$

$$C_x = \frac{X_j - X_i}{L} = 0$$

$$\left[R\right]^{(1)} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\left[K\right]^{(i)} = \frac{10(1)}{10} \begin{bmatrix} 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\left[K_{S}\right]^{(i)} = \left[R\right]^{T(i)} \left[K\right]^{(i)} \left[R\right]^{(i)}$$

$$\left[K_S\right]^{(1)} = \begin{bmatrix} 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

The stiffness matrices [K] for elements 2 and 3 are calculated in a similar fashion.

$$\left[K\right]^{(2)} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 \end{bmatrix}$$

$$\left[K\right]^{(3)} = \begin{bmatrix} B & -B & -B & B \\ -B & B & B & -B \\ -B & B & B & -B \\ B & -B & -B & B \end{bmatrix}$$

where $B = 1 / (2\sqrt{2})$

The structure stiffness matrix is then calculated from Equation (3.27)

$$[Ks] = \sum_{i=1}^{n} \left[R\right]^{T(i)} \left[K\right]^{(i)} \left[R\right]^{(i)}$$

$$\left[K_s\right] = \begin{bmatrix} 1 & 0 & -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & -1 \\ -1 & 0 & 1.354 & -.354 & -.354 & .354 \\ 0 & 0 & -.354 & .354 & .354 & -.354 \\ 0 & 0 & -.354 & .354 & .354 & -.354 \\ 0 & -1 & .354 & -.354 & -.354 & 1.354 \end{bmatrix}$$

From Equations (3.27) thru (3.30) the structure stiffness matrix is partitioned to form the submatrices $[K_{\alpha\alpha}]$ and $[k_{\alpha\beta}]^T$.

$$\begin{bmatrix} K_{\alpha\alpha} \end{bmatrix} = \begin{bmatrix} 1.354 & .354 \\ -.354 & .354 \end{bmatrix}$$

$$\begin{bmatrix} K_{\alpha\beta} \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & 0 \\ -.354 & .354 \\ .354 & -.354 \end{bmatrix}$$

and the unkown displacements and reactions are calculated

$$[\bar{S}_\alpha] = [K_{\alpha\alpha}]^{-1} [\bar{Q}_\alpha]$$

$$[\bar{Q}_\beta] = [K_{\alpha\beta}]^T [\bar{S}_\alpha]$$

$$\begin{bmatrix} S_\alpha \end{bmatrix} = \begin{bmatrix} 1.0 & 1.0 \\ 1.0 & 3.825 \end{bmatrix} \begin{bmatrix} 1.0 \\ 1.0 \end{bmatrix} \quad \begin{bmatrix} 2.000 \\ 4.825 \end{bmatrix}$$

$$\begin{bmatrix} Q_\beta \end{bmatrix} = \begin{bmatrix} -1.0 & 0. & 2.0 \\ 0.0 & 0. & \\ -.354 & 0.354 & 4.825 \\ 0.354 & -.354 & \end{bmatrix} \begin{bmatrix} -2.0 \\ 0.0 \\ 1.0 \\ -1 \end{bmatrix}$$

The element forces are then calculated

$$[Q] = [K]\ [R]\ [S]$$

For element 1

$$[Q]^{(1)} = \begin{bmatrix} 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 2.0 \\ 4.83 \end{bmatrix} \begin{bmatrix} -2.0 \\ 0.0 \\ 2.0 \\ 0.0 \end{bmatrix}$$

element forces for elements 2 and 3 are calculated using the same procedure.

Thus we obtain the deflections for the structure at node 2

X2=2.0 inches

Y2=4.8 inches

and the reactions at nodes 1 and 3 are

Rx1=-2.0 lb

Ry1= 0. lb

Rx3- 1.0 lb

Ry3=-1. lb

The element forces are calculated to be:

2.0 lb in tension for element 1,

0.0 lb for element 2, and

1.4 lb for element 3.

```
]RUN
        PROGRAM IASTRA

    ANALISING STRUCTURES WITH APPLE

THIS PROGRAM IS USED TO PREPARE THE INPUT DATA ?

CHOOSE ONE OF THE AVAILABLE OPTIONS:
    1. CREATE A NEW INPUT DATA FILE.
    2. LIST INPUT DATA.
    3. MODIFY AN EXISTING DATA FILE.
    4. COPY OPTION.
    5. EXIT FROM THE PROGRAM.
ENTER OPTION NUMBER ?1
SUBROUTINE HEADING
DWG ID CODE ?PR1
DATE ?24 OCT
YOUR NAME ?R CRESPO
ENTER THE FOLLOWING INFORMATION FOR EACH ELEMENT GENERATOR CARD IN FREE FORMAT.
NOTE:  ANY ERRORS NOT CORRECTED BEFORE  THE LINES ARE ENTERED CAN BE TAKEN CARE OF AFTER
HAVE BEEN ENTERED.

HIT RETURN TO CONTINUE
?
NUMBER OF ELEMENT GENERATING CARDS ?2
ENTER DATA FOR ELMT CARD # 1

1.  ELEMENT TYPE (OPTION #) ?1
2.  GROUP NUMBER?1
3.  NUMBER OF ELEMENTS ?2
4.  NODE #1?1
5.  INCREMENT #1 ?0
6.  NODE #2. ?2
7.  INCREMENT #2 ?1
ENTER DATA FOR ELMT CARD # 2

1.  ELEMENT TYPE (OPTION #) ?1
2.  GROUP NUMBER?2
3.  NUMBER OF ELEMENTS ?1
4.  NODE #1?2
6.  NODE #2. ?3
DO YOU WANT TO ADD MORE ELEMENT CARDS (Y OR N) ? N
ENTER MAX. NUMBER OF NODES (N<=50) ? 3
ENTER THE NPCO FOR NODE # 1
X COORDINATE ?0
Y COORDINATE ?0
Z COORDINATE ?0
ENTER THE NPCO FOR NODE # 2
X COORDINATE ?10
Y COORDINATE ?0
Z COORDINATE ?0
ENTER THE NPCO FOR NODE # 3
X COORDINATE ?0
Y COORDINATE ?10
Z COORDINATE ?0
DO YOU WANT TO ADD ANY MORE NODES (Y OR N) ? N
NUMBER OF SUPPRESSIONS CARDS ? 3
ENTER OPTION NUMBER
  1.  ONLY ROD ELEMENTS ARE USED.
  2.  A MIXTURE OF BEAM AND ROD ELEMENTS ARE USED.
```

```
ENTER OPTION NUMBER ? 1
ENTER NODE NUMBER AND Y OR N TO THE QUESTIONS.

NODAL SUPPRESSIONS CARD # ... 1
NODE NUMBER ? 1
SUPPRESS X ? Y
SUPPRESS Y ? Y
SUPPRESS Z ? Y
ENTER NODE NUMBER AND Y OR N TO THE QUESTIONS.

NODAL SUPPRESSIONS CARD # ... 2
NODE NUMBER ? 2
SUPPRESS X ? N
SUPPRESS Y ? N
SUPPRESS Z ? Y
ENTER NODE NUMBER AND Y OR N TO THE QUESTIONS.

NODAL SUPPRESSIONS CARD # ... 3
NODE NUMBER ? 3
SUPPRESS X ? Y
SUPPRESS Y ? Y
SUPPRESS Z ? Y
DO YOU WANT TO ADD ANY MORE SUPPRESSIONS ?(Y OR N) N
INPUT MAT. & SECT. PROPERTIES
NUMBER OF GROUPS ? 2
ENTER OPTION NUMBER
  1. ONLY ROD ELEMENTS ARE USED.
  2.  A MIXTURE OF RODS AND BEAM ELEMENTS ARE USED.
ENTER OPTION NUMBER ?1
MAT. & SECT. PROPERTIES FOR GROUP # 1
CROSS-SECTIONAL AREA ? 1

MODULUS OF ELASTICITY E ? 10
MAT. & SECT. PROPERTIES FOR GROUP # 2
CROSS-SECTIONAL AREA ? 1

MODULUS OF ELASTICITY E ? 10
DO YOU WANT TO ADD ANY MORE GROUPS ?(Y OR N) N
LOADS OPTION
THE FOLLOWING INPUT PERTAINS TO THE LOAD CARDS.
ENTER NUMBER OF LOAD CARDS ? 1
LOAD OPTIONS.
  1. ONLY ROD ELEMENTS ARE USED.
  2.  A MIXTURE OF RODS AND BEAM ELEMENTSS ARE USED.
ENTER LOAD OPTION ? 1
LOAD FOR LOAD CARD # 1
ENTER NODE NUMBER ? 2
ENTER F-X ? 1
ENTER F-Y ? 1
ENTER F-Z ? 0
DO YOU WANT TO ADD ANY MORE LOAD CARDS ?(Y OR N) N
INPUT SUBROUTINE COMPLETED
STORAGE OF INPUT DATA COMPLETE
         PROGRAM IASTRA

    ANALISING STRUCTURES WITH APPLE

THIS PROGRAM IS USED TO PREPARE THE INPUT DATA

CHOOSE ONE OF THE AVAILABLE OPTIONS:
```

```
    1. CREATE A NEW INPUT DATA FILE.
    2. LIST INPUT DATA.
    3. MODIFY AN EXISTING DATA FILE.
    4. COPY OPTION.
    5. EXIT FROM THE PROGRAM.
ENTER OPTION NUMBER ?5
END OF JOB
```

```
]RUN ASTRA1
        PROGRAM ASTRA1

    ANALISING STRUCTURES WITH APPLE

ENTER DRAWING CODE NUMBER ? RC1

READING INPUT INFORMATION
LIST INPUT INFORMATION

ELEMENT CARDS

  LINE ELMT  GROUP  # OF  NODE  INC  NODE  INC
   #   TYPE    #    ELMT   1     1    2     2

   1    1      1     2     1     0    2     1
   2    1      2     1     2     0    3     0


NODE #    X-COORD    Y-COORD    Z-COORD

1           0          0          0
2           10         0          0
3           0          10         0


LINE #  NODE #    X   Y   Z   RX   RY   RZ

  1       1       1   1   1   0    0    0
  2       2       0   0   1   0    0    0
  3       3       1   1   1   0    0    0


MATERIAL PROPERTIES

LINE # 1        2           3
GROUP  1        2           3
ELMT   1        1           0
AREA   1        1           0
I-YY   0        0           0
I-ZZ   0        0           0
E      10       10          0
V      0        0           0
J      0        0           0
X'     0        0           0
Y'     0        0           0
Z'     0        0           0


LOADS
LINE #  NODE #   PX      PY      PZ      MX      MY      MZ
  1       2      1       1       0       0       0       0

1/2 BANDWIDTH HAS BEEN CALCULATED AS (CF+1)*6= 18

SSM HAVE BEEN CLEARED

CREATING A NULL SSM
```

```
324              0              3
LAST ADDRESS OF SSM IS 324

NULL SSM HAS BEEN CREATED

SIZE OF SSM IS 18

ELEMENT # 1  NODE I= 1  NODE J= 2

THE DIRECTION COSINES CX,CY,CZ FOR ELEMENT 1 ARE
1   0   0
CLEARING BB,BC,BF,BH

ESM SAVED ON DISK
CALCULATION OF ELEMT. STIFF MATRIX (GLOBAL)
ASSY OF STRUCTURE STIFFNESS MATRIX
ELEMENT # 1 STIFFNESS HAS BEEN ADDED TO SSM

ELEMENT # 2  NODE I= 1  NODE J= 3

THE DIRECTION COSINES CX,CY,CZ FOR ELEMENT 2 ARE
0   1   0
CLEARING BB,BC,BF,BH

ESM SAVED ON DISK
CALCULATION OF ELEMT. STIFF MATRIX (GLOBAL)
ASSY OF STRUCTURE STIFFNESS MATRIX
ELEMENT # 2 STIFFNESS HAS BEEN ADDED TO SSM

ELEMENT # 3  NODE I= 2  NODE J= 3

THE DIRECTION COSINES CX,CY,CZ FOR ELEMENT 3 ARE
-.707106781   .707106781   0
CLEARING BB,BC,BF,BH

ESM SAVED ON DISK
CALCULATION OF ELEMT. STIFF MATRIX (GLOBAL)
ASSY OF STRUCTURE STIFFNESS MATRIX
ELEMENT # 3 STIFFNESS HAS BEEN ADDED TO SSM

ASSY OF SSM IS COMPLETE

SS HAS BEEN STORED IN DOFIU
P HAVE BEEN STORED IN DOFIU
DOFIU SAVED IN DISK


PROGRAM ASTRA1 IS COMPLETED.  TO CONTINUE LOAD AND RUN PROGRAM ASTRA2

]RUNLOAD ASTRA2
]RUN
         PROGRAM ASTRA2

READ DOFIU(BP,2)
DOFIU READ FROM DISK

READING INPUT INFORMATION
INPUT DATA HAS BEEN READ
DOFIU HAVE BEEN REARANGED
CLEARING MATRICES KA,KB,KC,LA,LB
```

```
THIS IS BV 2

U-A AND U-B HAVE BEEN CALCULATED

RANK OF CONSOLIDATED MATRIX K IS 9

RANK OF PARTITIONED MATRIX K-AA IS 2

RANK OF PARTITIONED MATRIX K-BB IS 7
MATRICES U-A AND U-B HAVE BEEN CALCULATED
BOFIU SAVED IN DISK

CHECK PRINTOUT OF BOFIU AND BU
1  1  0  1  1  7
2  1  0  1  2  8
3  1  0  1  3  1
4  0  0  0  7  2
5  0  0  0  8  3
6  0  0  0  9  9
7  1  1  0  13  13
8  1  1  0  14  14
9  1  0  1  15  15
10  0  0  0  0  0
11  0  0  0  0  0
12  0  0  0  0  0
13  1  0  1  0  0
14  1  0  1  0  0
15  1  0  1  0  0
16  0  0  0  0  0
17  0  0  0  0  0
18  0  0  0  0  0
BU SAVED IN DISK

ASSY OF PARTITIONED SSM K-AA

MATRIX K-AA STORED IN DISK
ASSY OF PARTITIONED SSM K-ABT

MATRIX K-ABT STORED IN DISK
CALCULATION OF LA,LB WHEN BV<=46 USING RAM

LA1 STORED IN DISK
DECOMPOSITION OF K INTO L IS COMPLETE
CALCULATE LB USING RAM

CALCULATION OF L INVERTED IS COMPLETE
RUN ASTRA3 TO COMPLETE THE JOB

]LOAD ASTRA3
]RUN

READING INPUT INFORMATION
BOFIU READ IN DISK
BU READ FROM DISK

CALCULATE KC USING RAM
TEST BV= 2
STORE KC1 IN DISK
```

```
K-A INVERSE HAS BEEN CALCULATED

CALCULATE NODAL DEFLECTIONS
CALCULATE REACTIONS
DEFLECTIONS HAVE BEEN STORED
PRINT NODAL LOAD, DEFLECTIONS AND REACTIONS

NODE #        DOF        NODE LOAD        DEFLECTIONS        REACTIONS

1             1          0                ****                 -2
1             2          0                ****                 0
1             3          0                ****                 0
1             4          0                0                  0
1             5          0                0                  0
1             6          0                0                  0

2             1          1                2                  0
2             2          1                4.82842714                 0
2             3          0                ****                 0
2             4          0                0                  0
2             5          0                0                  0
2             6          0                0                  0

3             1          0                ****                 1
3             2          0                ****                 -1
3             3          0                ****                 0
3             4          0                0                  0
3             5          0                0                  0
3             6          0                0                  0

STORE NODAL DEFLECTIONS
RUN ASTRA4 TO COMPLETE JOB

]LOAD ASTRA4
]RUN
PROGRAM ASTRA4


READING INPUT INFORMATION
READ NODAL DEFLECTIONS
CALCULATION OF ELEMENT FORCES

THE MEMBER FORCES WILL BE PRINTED OUT IN FREE FORMAT IN THE ORDER SHOWN BELOW.
FORMAT LIMITATION WITHIN THE BASIC PROGRAMMING LANGUAGE


ELEMT #  NODE #     PX          FY          FZ          MX          MY          MZ

1  1  -2  0  0  0  0  0
   2  2  0  0  0  0  0

2  1  0  0  0  0  0  0
   3  0  0  0  0  0  0

3  2  1.41421356  0  0  0  0  0
   3  -1.41421356  0  0  0  0  0

END OF JOB
```

```
]PR#6
] RUN ASTRA1
        PROGRAM ASTRA1

    ANALISING STRUCTURES WITH APPLE

ENTER DRAWING CODE NUMBER ? RC2

READING INPUT INFORMATION
LIST INPUT INFORMATION

ELEMENT CARDS

  LINE ELMT  GROUP  # OF  NODE  INC  NODE  INC
   #   TYPE   #     ELMT  1     1    2     2

   1    1     1     2     1     2    2     2
   2    1     2     2     3     1    1     1
   3    1     3     2     1     2    4     -2


NODE #   X-COORD    Y-COORD    Z-COORD

1         0          80         0
2         60         80         0
3         0          0          0
4         60         0          0


LINE #  NODE #   X   Y   Z   RX   RY   RZ

  1       1      0   0   1   0    0    0
  2       2      0   0   1   0    0    0
  3       3      1   1   1   0    0    0
  4       4      1   1   1   0    0    0


MATERIAL PROPERTIES

LINE # 1         2              3
GROUP  1         2              3
ELMT   1         1              1
AREA   6         8              10
I-YY   0         0              0
I-ZZ   0         0              0
E      10000     10000          10000
V      0         0              0
J      0         0              0
X'     0         0              0
Y'     0         0              0
Z'     0         0              0


LOADS
LINE #  NODE #   PX      PY      PZ      MX      MY      MZ
  1       2      20      10      0       0       0       0

1/2 BANDWIDTH HAS BEEN CALCULATED AS (CF+1)*6= 24
```

```
SSM HAVE BEEN CLEARED

CREATING A NULL SSM

576            0            4
LAST ADDRESS OF SSM IS 576

NULL SSM HAS BEEN CREATED

SIZE OF SSM IS 24

ELEMENT # 1  NODE I= 1  NODE J= 2

THE DIRECTION COSINES CX,CY,CZ FOR ELEMENT 1 ARE
.999999999   0   0
CLEARING BB,BC,BF,BH

ESM SAVED ON DISK
CALCULATION OF ELEMT. STIFF MATRIX (GLOBAL)
ASSY OF STRUCTURE STIFFNESS MATRIX
ELEMENT # 1 STIFFNESS HAS BEEN ADDED TO SSM

ELEMENT # 2  NODE I= 3  NODE J= 4

THE DIRECTION COSINES CX,CY,CZ FOR ELEMENT 2 ARE
.999999999   0   0
CLEARING BB,BC,BF,BH

ESM SAVED ON DISK
CALCULATION OF ELEMT. STIFF MATRIX (GLOBAL)
ASSY OF STRUCTURE STIFFNESS MATRIX
ELEMENT # 2 STIFFNESS HAS BEEN ADDED TO SSM

ELEMENT # 3  NODE I= 1  NODE J= 3

THE DIRECTION COSINES CX,CY,CZ FOR ELEMENT 3 ARE
0   -1   0
CLEARING BB,BC,BF,BH

ESM SAVED ON DISK
CALCULATION OF ELEMT. STIFF MATRIX (GLOBAL)
ASSY OF STRUCTURE STIFFNESS MATRIX
ELEMENT # 3 STIFFNESS HAS BEEN ADDED TO SSM

ELEMENT # 4  NODE I= 2  NODE J= 4

THE DIRECTION COSINES CX,CY,CZ FOR ELEMENT 4 ARE
0   -1   0
CLEARING BB,BC,BF,BH

ESM SAVED ON DISK
CALCULATION OF ELEMT. STIFF MATRIX (GLOBAL)
ASSY OF STRUCTURE STIFFNESS MATRIX
ELEMENT # 4 STIFFNESS HAS BEEN ADDED TO SSM

ELEMENT # 5  NODE I= 1  NODE J= 4

THE DIRECTION COSINES CX,CY,CZ FOR ELEMENT 5 ARE
.6   -.8   0
CLEARING BB,BC,BF,BH
```

```
ESM SAVED ON DISK
CALCULATION OF ELEMT. STIFF MATRIX (GLOBAL)
ASSY OF STRUCTURE STIFFNESS MATRIX
ELEMENT # 5 STIFFNESS HAS BEEN ADDED TO SSM

ELEMENT # 6  NODE I= 2  NODE J= 3

THE DIRECTION COSINES CX,CY,CZ FOR ELEMENT 6 ARE
-.6   -.8  0
CLEARING BB,BC,BF,BH

ESM SAVED ON DISK
CALCULATION OF ELEMT. STIFF MATRIX (GLOBAL)
ASSY OF STRUCTURE STIFFNESS MATRIX
ELEMENT # 6 STIFFNESS HAS BEEN ADDED TO SSM

ASSY OF SSM IS COMPLETE

SS HAS BEEN STORED IN DOFIU
F HAVE BEEN STORED IN DOFIU
DOFIU SAVED IN DISK


PROGRAM ASTRA1 IS COMPLETED.  TO CONTINUE LOAD AND RUN PROGRAM ASTRA2

]LOAD ASTRA2
]RUN
         PROGRAM ASTRA2

READ DOFIU(BF,2)
DOFIU READ FROM DISK

READING INPUT INFORMATION
INPUT DATA HAS BEEN READ
DOFIU HAVE BEEN REARANGED
CLEARING MATRICES KA,KB,KC,LA,LB
THIS IS BV 4

U-A AND U-B HAVE BEEN CALCULATED

RANK OF CONSOLIDATED MATRIX K IS 12

RANK OF PARTITIONED MATRIX K-AA IS 4

RANK OF PARTITIONED MATRIX K-BB IS 8
MATRICES U-A AND U-B HAVE BEEN CALCULATED
DOFIU SAVED IN DISK

CHECK PRINTOUT OF DOFIU AND BU
1  1  0  0  1  1
2  1  0  0  2  2
3  1  0  1  3  7
4  0  0  0  7  8
5  0  0  0  8  3
6  0  0  0  9  9
7  1  20  0  13  13
8  1  10  0  14  14
9  1  0  1  15  15
10  0  0  0  19  19
```

```
11  0  0  0  20  20
12  0  0  0  21  21
13  1  0  1  0  0
14  1  0  1  0  0
15  1  0  1  0  0
16  0  0  0  0  0
17  0  0  0  0  0
18  0  0  0  0  0
19  1  0  1  0  0
20  1  0  1  0  0
21  1  0  1  0  0
22  0  0  0  0  0
23  0  0  0  0  0
24  0  0  0  0  0
BU SAVED IN DISK

ASSY OF PARTITIONED SSM K-AA

MATRIX K-AA STORED IN DISK
ASSY OF PARTITIONED SSM K-ABT

MATRIX K-ABT STORED IN DISK
CALCULATION OF LA,LB WHEN BW<=48 USING RAM

LA1 STORED IN DISK
DECOMPOSITION OF K INTO L IS COMPLETE
CALCULATE LB USING RAM

CALCULATION OF L INVERTED IS COMPLETE
RUN ASTRA3 TO COMPLETE THE JOB

]LOAD ASTRA3
]RUN

READING INPUT INFORMATION
DOFIC READ IN DISK
BU READ FROM DISK

CALCULATE KC USING RAM
TEST BW= 4
STORE KC1 IN DISK


K-A INVERSE HAS BEEN CALCULATED

CALCULATE NODAL DEFLECTIONS
CALCULATE REACTIONS
DEFLECTIONS HAVE BEEN STORED
PRINT NODAL LOAD, DEFLECTIONS AND REACTIONS

NODE #          DOF         NODE LOAD        DEFLECTIONS          REACTIONS

1                1           0             .0350427345              0
1                2           0             .0102564101              0
1                3           0             ####                  0
1                4           0             0                  0
1                5           0             0                  0
1                6           0             0                  0

2                1           20            .0427350422              0
```

2	2	10	-6.41025625E-03	0
2	3	0	****	0
2	4	0	0	0
2	5	0	0	0
2	6	0	0	0
3	1	0	****	-12.3076922
3	2	0	****	-26.6666663
3	3	0	****	0
3	4	0	0	0
3	5	0	0	0
3	6	0	0	0
4	1	0	****	-7.69230756
4	2	0	****	16.6666663
4	3	0	****	0
4	4	0	0	0
4	5	0	0	0
4	6	0	0	0

```
STORE NODAL DEFLECTIONS
RUN ASTRA4 TO COMPLETE JOB

]LOAD ASTRA4
]RUN
PROGRAM ASTRA4


READING INPUT INFORMATION
READ NODAL DEFLECTIONS
CALCULATION OF ELEMENT FORCES

THE MEMBER FORCES WILL BE PRINTED OUT IN FREE FORMAT IN THE ORDER SHOWN BELOW.
FORMAT LIMITATION WITHIN THE BASIC PROGRAMMING LANGUAGE


ELEMT #  NODE #     FX          FY          FZ          MX          MY          MZ

1  1  -7.69230771  0  0  0  0  0
   2  7.69230771  0  0  0  0  0

2  3  0  0  0  0  0  0
   4  0  0  0  0  0  0

3  1  -10.2564101  0  0  0  0  0
   3  10.2564101  0  0  0  0  0

4  2  6.41025625  0  0  0  0  0
   4  -6.41025625  0  0  0  0  0

5  1  12.8205126  0  0  0  0  0
   4  -12.8205126  0  0  0  0  0

6  2  -20.5128203  0  0  0  0  0
   3  20.5128203  0  0  0  0  0

END OF JOB
```

```
]LOAD ASTRA1
]RUN
        PROGRAM ASTRA1

    ANALISING STRUCTURES WITH APPLE

ENTER DRAWING CODE NUMBER ? RC3

READING INPUT INFORMATION
LIST INPUT INFORMATION

ELEMENT CARDS

  LINE ELMT  GROUP  # OF  NODE  INC  NODE  INC
   #   TYPE    #    ELMT  1     1    2     2

   1    1      1     3    1     0    2     1
   2    1      1     2    2     0    3     1
   3    1      1     1    3     0    4     0


NODE #    X-COORD     Y-COORD     Z-COORD

1          0           0           0
2          75          0           0
3          0           100         0
4          0           0           100


LINE #  NODE #    X   Y    Z   RX    RY    RZ

  1       1       1   1    1   0     0     0
  2       2       0   1    1   0     0     0
  3       3       1   1    1   0     0     0
  4       4       0   0    0   0     0     0


MATERIAL PROPERTIES

LINE # 1         2               3
GROUP  1         2               3
ELMT   1         0               0
AREA   10        0               0
I-YY   0         0               0
I-ZZ   0         0               0
E      10000     0               0
V      0         0               0
J      0         0               0
X'     0         0               0
Y'     0         0               0
Z'     0         0               0


LOADS
LINE #  NODE #   FX       PY        PZ        MX        MY        MZ
  1       4      30     40      0        0         0         0

1/2 BANDWIDTH HAS BEEN CALCULATED AS (CF+1)*6= 24
```

```
SSM HAVE BEEN CLEARED

CREATING A NULL SSM

576            0              4
LAST ADDRESS OF SSM IS 576

NULL SSM HAS BEEN CREATED

SIZE OF SSM IS 24

ELEMENT # 1  NODE I= 1  NODE J= 2

THE DIRECTION COSINES CX,CY,CZ FOR ELEMENT 1 ARE
.999999999   0   0
CLEARING BB,BC,BF,BH

ESM SAVED ON DISK
CALCULATION OF ELEMT. STIFF MATRIX (GLOBAL)
ASSY OF STRUCTURE STIFFNESS MATRIX
ELEMENT # 1 STIFFNESS HAS BEEN ADDED TO SSM

ELEMENT # 2  NODE I= 1  NODE J= 3

THE DIRECTION COSINES CX,CY,CZ FOR ELEMENT 2 ARE
0   .999999999   0
CLEARING BB,BC,BF,BH

ESM SAVED ON DISK
CALCULATION OF ELEMT. STIFF MATRIX (GLOBAL)
ASSY OF STRUCTURE STIFFNESS MATRIX
ELEMENT # 2 STIFFNESS HAS BEEN ADDED TO SSM

ELEMENT # 3  NODE I= 1  NODE J= 4

THE DIRECTION COSINES CX,CY,CZ FOR ELEMENT 3 ARE
0   0   .999999999
CLEARING BB,BC,BF,BH

ESM SAVED ON DISK
CALCULATION OF ELEMT. STIFF MATRIX (GLOBAL)
ASSY OF STRUCTURE STIFFNESS MATRIX
ELEMENT # 3 STIFFNESS HAS BEEN ADDED TO SSM

ELEMENT # 4  NODE I= 2  NODE J= 3

THE DIRECTION COSINES CX,CY,CZ FOR ELEMENT 4 ARE
-.6   .799999999   0
CLEARING BB,BC,BF,BH

ESM SAVED ON DISK
CALCULATION OF ELEMT. STIFF MATRIX (GLOBAL)
ASSY OF STRUCTURE STIFFNESS MATRIX
ELEMENT # 4 STIFFNESS HAS BEEN ADDED TO SSM

ELEMENT # 5  NODE I= 2  NODE J= 4

THE DIRECTION COSINES CX,CY,CZ FOR ELEMENT 5 ARE
-.6   0   .799999999
CLEARING BB,BC,BF,BH
```

```
ESM SAVED ON DISK
CALCULATION OF ELEMT. STIFF MATRIX (GLOBAL)
ASSY OF STRUCTURE STIFFNESS MATRIX
ELEMENT # 5 STIFFNESS HAS BEEN ADDED TO SSM

ELEMENT # 6  NODE I= 3  NODE J= 4

THE DIRECTION COSINES CX,CY,CZ FOR ELEMENT 6 ARE
0   -.707106781    .707106781
CLEARING BB,BC,BF,BH

ESM SAVED ON DISK
CALCULATION OF ELEMT. STIFF MATRIX (GLOBAL)
ASSY OF STRUCTURE STIFFNESS MATRIX
ELEMENT # 6 STIFFNESS HAS BEEN ADDED TO SSM

ASSY OF SSM IS COMPLETE

SS HAS BEEN STORED IN DOFIU
P HAVE BEEN STORED IN DOFIU
DOFIU SAVED IN DISK


PROGRAM ASTRA1 IS COMPLETED.  TO CONTINUE LOAD AND RUN PROGRAM ASTRA2

]LOAD ASTRA 22
]RUN
         PROGRAM ASTRA2

READ DOFIU(BF,2)
DOFIU READ FROM DISK

READING INPUT INFORMATION
INPUT DATA HAS BEEN READ
DOFIU HAVE BEEN REARANGED
CLEARING MATRICES KA,KB,KC,LA,LB
THIS IS BV 4

U-A AND U-B HAVE BEEN CALCULATED

RANK OF CONSOLIDATED MATRIX K IS 12

RANK OF PARTITIONED MATRIX K-AA IS 4

RANK OF PARTITIONED MATRIX K-BB IS 8
MATRICES U-A AND U-B HAVE BEEN CALCULATED
DOFIU SAVED IN DISK

CHECK PRINTOUT OF DOFIU AND BU
1  1  0  1  1  7
2  1  0  1  2  19
3  1  0  1  3  20
4  0  0  0  7  21
5  0  0  0  8  1
6  0  0  0  9  2
7  1  0  0  13  3
8  1  0  1  14  8
9  1  0  1  15  9
10  0  0  0  19  13
```

```
11  0  0  0  20  14
12  0  0  0  21  15
13  1  0  1  0  0
14  1  0  1  0  0
15  1  0  1  0  0
16  0  0  0  0  0
17  0  0  0  0  0
18  0  0  0  0  0
19  1  30  0  0  0
20  1  40  0  0  0
21  1  0  0  0  0
22  0  0  0  0  0
23  0  0  0  0  0
24  0  0  0  0  0
BU SAVED IN DISK

ASSY OF PARTITIONED SSM K-AA

MATRIX K-AA STORED IN DISK
ASSY OF PARTITIONED SSM K-ABT

MATRIX K-ABT STORED IN DISK
CALCULATION OF LA,LB WHEN BV<=48 USING RAM

LA1 STORED IN DISK
DECOMPOSITION OF K INTO L IS COMPLETE
CALCULATE LB USING RAM

CALCULATION OF L INVERTED IS COMPLETE
RUN ASTRA3 TO COMPLETE THE JOB

]LOAD ASTRA3
]RUN

READING INPUT INFORMATION
DOFID READ IN DISK
BU READ FROM DISK

CALCULATE KC USING RAM
TEST BV= 4
STORE KC1 IN DISK


K-A INVERSE HAS BEEN CALCULATED

CALCULATE NODAL DEFLECTIONS
CALCULATE REACTIONS
DEFLECTIONS HAVE BEEN STORED
PRINT NODAL LOAD, DEFLECTIONS AND REACTIONS
```

NODE #	DOF	NODE LOAD	DEFLECTIONS	REACTIONS
1	1	0	****	-24.6710526
1	2	0	****	0
1	3	0	****	-79.9999998
1	4	0	0	0
1	5	0	0	0
1	6	0	0	0
2	1	0	.0165032695	0

```
2            2         0         ****              -7.10526315
2            3         0         ****              39.9999997
2            4         0         0            0
2            5         0         0            0
2            6         0         0            0

3            1         0         ****              -5.32894737
3            2         0         ****              -32.8947369
3            3         0         ****              40
3            4         0         0            0
3            5         0         0            0
3            6         0         0            0

4            1         30        .229336622            0
4            2         40        .193137085            0
4            3         0         .0799999999            0
4            4         0         0            0
4            5         0         0            0
4            6         0         0            0

STORE NODAL DEFLECTIONS
RUN ASTRA4 TO COMPLETE JOB

]LOAD ASTRA4
]RUN
PROGRAM ASTRA4


READING INPUT INFORMATION
READ NODAL DEFLECTIONS
CALCULATION OF ELEMENT FORCES

THE MEMBER FORCES WILL BE PRINTED OUT IN FREE FORMAT IN THE ORDER SHOWN BELOW.
FORMAT LIMITATION WITHIN THE BASIC PROGRAMMING LANGUAGE


ELEMT #  NODE #     PX          PY          PZ          MX          MY          MZ

1  1  -24.6710526  0  0  0  0  0
   2  24.6710526  0  0  0  0  0

2  1  0  0  0  0  0  0
   3  0  0  0  0  0  0

3  1  -79.9999999  0  0  0  0  0
   4  79.9999999  0  0  0  0  0

4  2  -8.88157897  0  0  0  0  0
   3  8.88157897  0  0  0  0  0

5  2  49.9999993  0  0  0  0  0
   4  -49.9999998  0  0  0  0  0

6  3  56.5685425  0  0  0  0  0
   4  -56.5685425  0  0  0  0  0

END OF JOB
```

```
]LOAD ASTRA1
]RUN
        PROGRAM ASTRA1

    ANALISING STRUCTURES WITH APPLE

ENTER DRAWING CODE NUMBER ? RC4

READING INPUT INFORMATION
LIST INPUT INFORMATION

ELEMENT CARDS

  LINE ELMT  GROUP  # OF  NODE  INC  NODE  INC
   #   TYPE    #    ELMT  1     1    2     2

   1    2      1     2     4    1     5     1
   2    2      1     2     7    1     8     1
   3    2      1     1    10    0    11     0
   4    2      2     3     1    3     4     3
   5    2      2     3     2    3     5     3
   6    2      2     2     3    3     6     3

NODE #   X-COORD    Y-COORD    Z-COORD

1          0          0          0
2          240        0          0
3          480        0          0
4          0          144        0
5          240        144        0
6          480        144        0
7          0          288        0
8          240        288        0
9          480        288        0
10         0          528        0
11         240        528        0

LINE #  NODE #   X   Y   Z   RX   RY   RZ

  1      1       1   1   1   1    1    1
  2      2       0   1   1   1    1    0
  3      3       1   1   1   1    1    0
  4      4       0   0   1   1    1    0
  5      5       0   0   1   1    1    0
  6      6       0   0   1   1    1    0
  7      7       0   0   1   1    1    0
  8      8       0   0   1   1    1    0
  9      9       0   0   1   1    1    0
  10     10      0   0   1   1    1    0
  11     11      0   0   1   1    1    0

MATERIAL PROPERTIES

LINE # 1         2                3
GROUP  1         2                3
ELMT   2         2                0
AREA   27.7      14.7             0
```

I-YY	124	56.4	0
I-ZZ	3270	395	0
E	29000000	29000000	
0			
V	.25	.25	0
J	50	50	0
X'	0	-900	0
Y'	900	0	0
Z'	0	0	0

LOADS

LINE #	NODE #	PX	PY	PZ	MX	MY	MZ
1	4	50000	-50000	0	0	0	-2000000
2	5	0	-100000	0	0	0	0
3	6	0	-50000	0	0	0	2000000
4	7	75000	-50000	0	0	0	-2000000
5	8	0	-150000	0	0	0	-2000000
6	9	0	-100000	0	0	0	4000000
7	10	100000	-100000	0	0	0	-4000000
8	11	0	-100000	0	0	0	4000000

```
1/2 BANDWIDTH HAS BEEN CALCULATED AS (CF+1)*6= 24

SSM HAVE BEEN CLEARED

CREATING A NULL SSM

1584            0              11
LAST ADDRESS OF SSM IS 1584

NULL SSM HAS BEEN CREATED

SIZE OF SSM IS 66

ELEMENT # 1  NODE I= 4  NODE J= 5

THE DIRECTION COSINES CX,CY,CZ FOR ELEMENT 1 ARE
1   0   0

THE FOLLOWING ARE THE DIRECTION COSINES CG,SG .999999998   0

CLEARING BB,BC,BF,BH

LOCAL STIFF. OF ELMT 1 IS COMPLETED

ESM SAVED ON DISK
CALCULATION OF ELEMT. STIFF MATRIX (GLOBAL)
ASSY OF STRUCTURE STIFFNESS MATRIX
ELEMENT #.1 STIFFNESS HAS BEEN ADDED TO SSM

ELEMENT # 2  NODE I= 5  NODE J= 6

THE DIRECTION COSINES CX,CY,CZ FOR ELEMENT 2 ARE
1   0   0

THE FOLLOWING ARE THE DIRECTION COSINES CG,SG .999999998   0

CLEARING BB,BC,BF,BH
```

```
LOCAL STIFF. OF ELMT 2 IS COMPLETED

ESM SAVED ON DISK
CALCULATION OF ELEMT. STIFF MATRIX (GLOBAL)
ASSY OF STRUCTURE STIFFNESS MATRIX
ELEMENT # 2 STIFFNESS HAS BEEN ADDED TO SSM

ELEMENT # 3  NODE I= 7  NODE J= 8

THE DIRECTION COSINES CX,CY,CZ FOR ELEMENT 3 ARE
1   0   0

THE FOLLOWING ARE THE DIRECTION COSINES CG,SG .999999998   0

CLEARING BB,BC,BF,BH

LOCAL STIFF. OF ELMT 3 IS COMPLETED

ESM SAVED ON DISK
CALCULATION OF ELEMT. STIFF MATRIX (GLOBAL)
ASSY OF STRUCTURE STIFFNESS MATRIX
ELEMENT # 3 STIFFNESS HAS BEEN ADDED TO SSM

ELEMENT # 4  NODE I= 8  NODE J= 9

THE DIRECTION COSINES CX,CY,CZ FOR ELEMENT 4 ARE
1   0   0

THE FOLLOWING ARE THE DIRECTION COSINES CG,SG .999999998   0

CLEARING BB,BC,BF,BH

LOCAL STIFF. OF ELMT 4 IS COMPLETED

ESM SAVED ON DISK
CALCULATION OF ELEMT. STIFF MATRIX (GLOBAL)
ASSY OF STRUCTURE STIFFNESS MATRIX
ELEMENT # 4 STIFFNESS HAS BEEN ADDED TO SSM

ELEMENT # 5  NODE I= 10  NODE J= 11

THE DIRECTION COSINES CX,CY,CZ FOR ELEMENT 5 ARE
1   0   0

THE FOLLOWING ARE THE DIRECTION COSINES CG,SG .999999998   0

CLEARING BB,BC,BF,BH

LOCAL STIFF. OF ELMT 5 IS COMPLETED

ESM SAVED ON DISK
CALCULATION OF ELEMT. STIFF MATRIX (GLOBAL)
ASSY OF STRUCTURE STIFFNESS MATRIX
ELEMENT # 5 STIFFNESS HAS BEEN ADDED TO SSM

ELEMENT # 6  NODE I= 1  NODE J= 4

THE DIRECTION COSINES CX,CY,CZ FOR ELEMENT 6 ARE
0   1   0
```

```
THE FOLLOWING ARE THE DIRECTION COSINES CG,SG .999999998   0

CLEARING BB,BC,BF,BH

LOCAL STIFF. OF ELMT 6 IS COMPLETED

ESM SAVED ON DISK
CALCULATION OF ELEMT. STIFF MATRIX (GLOBAL)
ASSY OF STRUCTURE STIFFNESS MATRIX
ELEMENT # 6 STIFFNESS HAS BEEN ADDED TO SSM

ELEMENT # 7  NODE I= 4  NODE J= 7

THE DIRECTION COSINES CX,CY,CZ FOR ELEMENT 7 ARE
0   1   0

THE FOLLOWING ARE THE DIRECTION COSINES CG,SG .999999998   0

CLEARING BB,BC,BF,BH

LOCAL STIFF. OF ELMT 7 IS COMPLETED

ESM SAVED ON DISK
CALCULATION OF ELEMT. STIFF MATRIX (GLOBAL)
ASSY OF STRUCTURE STIFFNESS MATRIX
ELEMENT # 7 STIFFNESS HAS BEEN ADDED TO SSM

ELEMENT # 8  NODE I= 7  NODE J= 10

THE DIRECTION COSINES CX,CY,CZ FOR ELEMENT 8 ARE
0   1   0

THE FOLLOWING ARE THE DIRECTION COSINES CG,SG .999999998   0

CLEARING BB,BC,BF,BH

LOCAL STIFF. OF ELMT 8 IS COMPLETED

ESM SAVED ON DISK
CALCULATION OF ELEMT. STIFF MATRIX (GLOBAL)
ASSY OF STRUCTURE STIFFNESS MATRIX
ELEMENT # 8 STIFFNESS HAS BEEN ADDED TO SSM

ELEMENT # 9  NODE I= 2  NODE J= 5

THE DIRECTION COSINES CX,CY,CZ FOR ELEMENT 9 ARE
0   1   0

THE FOLLOWING ARE THE DIRECTION COSINES CG,SG .999999998   0

CLEARING BB,BC,BF,BH

LOCAL STIFF. OF ELMT 9 IS COMPLETED

ESM SAVED ON DISK
CALCULATION OF ELEMT. STIFF MATRIX (GLOBAL)
ASSY OF STRUCTURE STIFFNESS MATRIX
ELEMENT # 9 STIFFNESS HAS BEEN ADDED TO SSM
```

```
ELEMENT # 10  NODE I= 5  NODE J= 8

THE DIRECTION COSINES CX,CY,CZ FOR ELEMENT 10 ARE
0   1   0

THE FOLLOWING ARE THE DIRECTION COSINES CG,SG .999999998   0

CLEARING BB,BC,BF,BH

LOCAL STIFF. OF ELMT 10 IS COMPLETED

ESM SAVED ON DISK
CALCULATION OF ELEMT. STIFF MATRIX (GLOBAL)
ASSY OF STRUCTURE STIFFNESS MATRIX
ELEMENT # 10 STIFFNESS HAS BEEN ADDED TO SSM

ELEMENT # 11  NODE I= 8  NODE J= 11

THE DIRECTION COSINES CX,CY,CZ FOR ELEMENT 11 ARE
0   1   0

THE FOLLOWING ARE THE DIRECTION COSINES CG,SG .999999998   0

CLEARING BB,BC,BF,BH

LOCAL STIFF. OF ELMT 11 IS COMPLETED

ESM SAVED ON DISK
CALCULATION OF ELEMT. STIFF MATRIX (GLOBAL)
ASSY OF STRUCTURE STIFFNESS MATRIX
ELEMENT # 11 STIFFNESS HAS BEEN ADDED TO SSM

ELEMENT # 12  NODE I= 3  NODE J= 6

THE DIRECTION COSINES CX,CY,CZ FOR ELEMENT 12 ARE
0   1   0

THE FOLLOWING ARE THE DIRECTION COSINES CG,SG .999999999   0

CLEARING BB,BC,BF,BH

LOCAL STIFF. OF ELMT 12 IS COMPLETED

ESM SAVED ON DISK
CALCULATION OF ELEMT. STIFF MATRIX (GLOBAL)
ASSY OF STRUCTURE STIFFNESS MATRIX
ELEMENT # 12 STIFFNESS HAS BEEN ADDED TO SSM

ELEMENT # 13  NODE I= 6  NODE J= 9

THE DIRECTION COSINES CX,CY,CZ FOR ELEMENT 13 ARE
0   1   0

THE FOLLOWING ARE THE DIRECTION COSINES CG,SG .999999999   0

CLEARING BB,BC,BF,BH

LOCAL STIFF. OF ELMT 13 IS COMPLETED

ESM SAVED ON DISK
```

```
CALCULATION OF ELEMT. STIFF MATRIX (GLOBAL)
ASSY OF STRUCTURE STIFFNESS MATRIX
ELEMENT # 13 STIFFNESS HAS BEEN ADDED TO SSM

ASSY OF SSM IS COMPLETE

SS HAS BEEN STORED IN DOFIU
P HAVE BEEN STORED IN DOFIU
DOFIU SAVED IN DISK


PROGRAM ASTRA1 IS COMPLETED.  TO CONTINUE LOAD AND RUN PROGRAM ASTRA2

]LOAD ASTRA2
]RUN
         PROGRAM ASTRA2

READ DOFIU(BF,2)
DOFIU READ FROM DISK

READING INPUT INFORMATION
INPUT DATA HAS BEEN READ
DOFIU HAVE BEEN REARANGED
CLEARING MATRICES KA,KB,KC,LA,LB
THIS IS BW 27

U-A AND U-B HAVE BEEN CALCULATED

RANK OF CONSOLIDATED MATRIX K IS 66

RANK OF PARTITIONED MATRIX K-AA IS 27

RANK OF PARTITIONED MATRIX K-BB IS 39
MATRICES U-A AND U-B HAVE BEEN CALCULATED
DOFIU SAVED IN DISK

CHECK PRINTOUT OF DOFIU AND BW
1  1  0  1  1  7
2  1  0  1  2  12
3  1  0  1  3  18
4  1  0  1  4  19
5  1  0  1  5  20
6  1  0  1  6  24
7  1  0  0  7  25
8  1  0  1  8  26
9  1  0  1  9  30
10  1  0  1  10  31
11  1  0  1  11  32
12  1  0  0  12  36
13  1  0  1  13  37
14  1  0 -1  14  38
15  1  0  1  15  42
16  1  0  1  16  43
17  1  0  1  17  44
18  1  0  0  18  48
19  1  50000  0  19  49
20  1  -50000  0  20  50
21  1  0  1  21  54
22  1  0  1  22  55
23  1  0  1  23  56
```

```
24  1  -2000000  0  24  60
25  1  0  0  25  61
26  1  -100000  0  26  62
27  1  0  1  27  66
28  1  0  1  28  1
29  1  0  1  29  2
30  1  0  0  30  3
31  1  0  0  31  4
32  1  -50000  0  32  5
33  1  0  1  33  6
34  1  0  1  34  8
35  1  0  1  35  9
36  1  2000000  0  36  10
37  1  75000  0  37  11
38  1  -50000  0  38  13
39  1  0  1  39  14
40  1  0  1  40  15
41  1  0  1  41  16
42  1  -2000000  0  42  17
43  1  0  0  43  21
44  1  -150000  0  44  22
45  1  0  1  45  23
46  1  0  1  46  27
47  1  0  1  47  28
48  1  -2000000  0  48  29
49  1  0  0  49  33
50  1  -100000  0  50  34
51  1  0  1  51  35
52  1  0  1  52  39
53  1  0  1  53  40
54  1  4000000  0  54  41
55  1  100000  0  55  45
56  1  -100000  0  56  46
57  1  0  1  57  47
58  1  0  1  58  51
59  1  0  1  59  52
60  1  -4000000  0  60  53
61  1  0  0  61  57
62  1  -100000  0  62  58
63  1  0  1  63  59
64  1  0  1  64  63
65  1  0  1  65  64
66  1  4000000  0  66  65
BC SAVED IN DISK

ASSY OF PARTITIONED SSM K-AA

MATRIX K-AA STORED IN DISK
ASSY OF PARTITIONED SSM K-ABT

MATRIX K-ABT STORED IN DISK
CALCULATION OF LA,LB WHEN BW<=48 USING RAM

LA1 STORED IN DISK
DECOMPOSITION OF K INTO L IS COMPLETE
CALCULATE LB USING RAM

CALCULATION OF L INVERTED IS COMPLETE
RUN ASTRAS TO COMPLETE THE JOB
```

```
]LOAD ASTRA3
]RUN

READING INPUT INFORMATION
DOFID READ IN DISK
BC READ FROM DISK

CALCULATE KC USING RAM
TEST BV= 27
STORE KC1 IN DISK


K-A INVERSE HAS BEEN CALCULATED

CALCULATE NODAL DEFLECTIONS
CALCULATE REACTIONS
DEFLECTIONS HAVE BEEN STORED
PRINT NODAL LOAD, DEFLECTIONS AND REACTIONS
```

NODE #	DOF	NODE LOAD	DEFLECTIONS	REACTIONS
1	1	0	****	-179910.986
1	2	0	****	-5805.72102
1	3	0	****	0
1	4	0	****	0
1	5	0	****	0
1	6	0	****	13906657.9
2	1	0	5.13733154	0
2	2	0	****	479555.921
2	3	0	****	0
2	4	0	****	0
2	5	0	****	0
2	6	0	2.33220186E-03	0
3	1	0	****	-45088.9665
3	2	0	****	226249.795
3	3	0	****	0
3	4	0	****	0
3	5	0	****	0
3	6	0	-.0469954646	0
4	1	50000	4.77075584	0
4	2	-50000	1.96111618E-03	0
4	3	0	****	0
4	4	0	****	0
4	5	0	****	0
4	6	-2000000	-.0119609366	0
5	1	0	4.80149445	0
5	2	-100000	-.161989334	0
5	3	0	****	0
5	4	0	****	0
5	5	0	****	0
5	6	0	2.33220209E-03	0
6	1	0	4.80845066	0
6	2	-50000	-.0764249836	0
6	3	0	****	0
6	4	0	****	0

```
6      5      0            ****                 0
6      6      2000000      -6.1851267E-03                 0

7      1      75000        6.65488978                 0
7      2      -50000       -6.82469405E-03                0
7      3      0            ****                 0
7      4      0            ****                 0
7      5      0            ****                 0
7      6      -2000000     -6.03368303E-03                0

8      1      0            6.62746454                 0
8      2      -150000      -.277798598                 0
8      3      0            ****                 0
8      4      0            ****                 0
8      5      0            ****                 0
8      6      -2000000     -3.67690258E-03                0

9      1      0            6.6070372                 0
9      2      -100000      -.120725053                 0
9      3      0            ****                 0
9      4      0            ****                 0
9      5      0            ****                 0
9      6      4000000      1.8327541E-03                 0

10     1      100000       12.7303939                 0
10     2      -100000      -.0347633772                0
10     3      0            ****                 0
10     4      0            ****                 0
10     5      0            ****                 0
10     6      -4000000     -7.87082248E-03                0

11     1      0            12.7136094                 0
11     2      -100000      -.362456679                 0
11     3      0            ****                 0
11     4      0            ****                 0
11     5      0            ****                 0
11     6      4000000      4.05028969E-05                0

STORE NODAL DEFLECTIONS
RUN ASTRA4 TO COMPLETE JOB

]LOAD ASTRA4
]RUN
PROGRAM ASTRA4


READING INPUT INFORMATION
READ NODAL DEFLECTIONS
CALCULATION OF ELEMENT FORCES

THE MEMBER FORCES WILL BE PRINTED OUT IN FREE FORMAT IN THE ORDER SHOWN BELOW.
FORMAT LIMITATION WITHIN THE BASIC PROGRAMMING LANGUAGE


ELEMT #  NODE #     PX          PY           PZ          MX          MY          MZ

1  4  -102884.687  -81815.38  0  0  0  -15473324.5
   5  102884.687  81815.38  0  0  0  -4162366.71
```

```
2  5  -23283.084  -45103.132  0  0  0  -2046966.32
   6  23283.084  45103.132  0  0  0  -8777785.38

3  7  91794.5587  -73616.4274  0  0  0  -9765194.18
   8  -91794.5587  73616.4274  0  0  0  -7902748.44

4  8  68372.0075  -31146.6633  0  0  0  -5914602.7
   9  -68372.0075  31146.6633  0  0  0  -1560596.51

5  10  56179.0959  -50373.9138  0  0  0  -9170832.11
   11  -56179.0959  50373.9138  0  0  0  -2918907.24

6  1  -5805.72101  179910.987  0  0  0  13906657.9
   4  5805.72101  -179910.987  0  0  0  12000524.2

7  4  26009.659  27026.3331  0  0  0  1472800.25
   7  -26009.659  -27026.3331  0  0  0  2418991.76

8  7  49626.086  43820.9776  0  0  0  5346202.41
   10  -49626.086  -43820.9776  0  0  0  5170832.14

9  2  479555.923  -1.2409687E-04  0  0  0  -.0361375809
   5  -479555.923  1.2409687E-04  0  0  0  9.67025757E-04

10  5  342843.674  79601.6273  0  0  0  .6209333.11
    8  -342843.674  -79601.6273  0  0  0  5253301.25

11  8  150373.917  56178.988  0  0  0  6564049.86
    11  -150373.917  -56178.988  0  0  0  6918907.19

12  3  226249.795  45088.967  0  0  0  -.0222396851
    6  -226249.795  -45088.967  0  0  0  6492811.36

13  6  131146.664  68372.0168  0  0  0  4284973.94
    9  -131146.664  -68372.0168  0  0  0  5560596.5

END OF JOB
```

```
]LOADRUN ASTRA1
        PROGRAM ASTRA1

    ANALISING STRUCTURES WITH APPLE

ENTER DRAWING CODE NUMBER ? RC5

READING INPUT INFORMATION
LIST INPUT INFORMATION

ELEMENT CARDS

  LINE ELMT  GROUP  # OF  NODE  INC  NODE  INC
   #   TYPE    #    ELMT  1     1    2     2

   1   2       1    1     1     0    4     0
   2   2       1    1     2     0    4     0
   3   2       1    1     3     0    4     0


NODE #   X-COORD    Y-COORD    Z-COORD

1         0         0          0
2         60        0          120
3         120       0          0
4         60        60         60


LINE #  NODE #   X   Y   Z   RX   RY   RZ

  1      1       1   1   1   1    1    1
  2      2       1   1   1   1    1    1
  3      3       1   1   1   1    1    1


MATERIAL PROPERTIES

LINE # 1        2              3
GROUP  1        2              3
ELMT   2        0              0
AREA   30       0              0
I-YY   20       0              0
I-ZZ   30       0              0
E      29000000 0              0
V      .25      0              0
J      50       0              0
X'     0        0              0
Y'     0        0              0
Z'     1000     0              0


LOADS
LINE #  NODE #   PX      PY       PZ       MX       MY       MZ
  1      4       10000  0        0        0         0        0

1/2 BANDWIDTH HAS BEEN CALCULATED AS (CF+1)*6= 24

SSM HAVE BEEN CLEARED
```

```
CREATING A NULL SSM

576            0            4
LAST ADDRESS OF SSM IS 576

NULL SSM HAS BEEN CREATED

SIZE OF SSM IS 24

ELEMENT # 1  NODE I= 1  NODE J= 4

THE DIRECTION COSINES CX,CY,CZ FOR ELEMENT 1 ARE
.577350269   .577350269   .577350269

THE FOLLOWING ARE THE DIRECTION COSINES CG,SG -.499999999   .866025403

CLEARING BB,BC,BF,BH

LOCAL STIFF. OF ELMT 1 IS COMPLETED

ESM SAVED ON DISK
CALCULATION OF ELEMT. STIFF MATRIX (GLOBAL)
ASSY OF STRUCTURE STIFFNESS MATRIX
ELEMENT # 1 STIFFNESS HAS BEEN ADDED TO SSM

ELEMENT # 2  NODE I= 2  NODE J= 4

THE DIRECTION COSINES CX,CY,CZ FOR ELEMENT 2 ARE
0   .707106781   -.707106781

THE FOLLOWING ARE THE DIRECTION COSINES CG,SG .995383406   -.0959785031

CLEARING BB,BC,BF,BH

LOCAL STIFF. OF ELMT 2 IS COMPLETED

ESM SAVED ON DISK
CALCULATION OF ELEMT. STIFF MATRIX (GLOBAL)
ASSY OF STRUCTURE STIFFNESS MATRIX
ELEMENT # 2 STIFFNESS HAS BEEN ADDED TO SSM

ELEMENT # 3  NODE I= 3  NODE J= 4

THE DIRECTION COSINES CX,CY,CZ FOR ELEMENT 3 ARE
-.577350269   .577350269   .577350269

THE FOLLOWING ARE THE DIRECTION COSINES CG,SG -.592136907   -.805837378

CLEARING BB,BC,BF,BH

LOCAL STIFF. OF ELMT 3 IS COMPLETED

ESM SAVED ON DISK
CALCULATION OF ELEMT. STIFF MATRIX (GLOBAL)
ASSY OF STRUCTURE STIFFNESS MATRIX
ELEMENT # 3 STIFFNESS HAS BEEN ADDED TO SSM

ASSY OF SSM IS COMPLETE

SS HAS BEEN STORED IN DOFIU
```

```
P HAVE BEEN STORED IN DOFIU
DOFIU SAVED IN DISK


PROGRAM ASTRA1 IS COMPLETED.  TO CONTINUE LOAD AND RUN PROGRAM ASTRA2

]LOAD ASTRA2
]RIUN
        PROGRAM ASTRA2

READ DOFIU(BP,2)
DOFIU READ FROM DISK

READING INPUT INFORMATION
INPUT DATA HAS BEEN READ
DOFIU HAVE BEEN REARANGED
CLEARING MATRICES KA,KB,KC,LA,LB
THIS IS BV 6

U-A AND U-B HAVE BEEN CALCULATED

RANK OF CONSOLIDATED MATRIX K IS 24

RANK OF PARTITIONED MATRIX K-AA IS 6

RANK OF PARTITIONED MATRIX K-BB IS 18
MATRICES U-A AND U-B HAVE BEEN CALCULATED
DOFIU SAVED IN DISK

CHECK PRINTOUT OF DOFIU AND BU
1  1  0  1  1  19
2  1  0  1  2  20
3  1  0  1  3  21
4  1  0  1  4  22
5  1  0  1  5  23
6  1  0  1  6  24
7  1  0  1  7  1
8  1  0  1  8  2
9  1  0  1  9  3
10  1  0  1  10  4
11  1  0  1  11  5
12  1  0  1  12  6
13  1  0  1  13  7
14  1  0  1  14  8
15  1  0  1  15  9
16  1  0  1  16  10
17  1  0  1  17  11
18  1  0  1  18  12
19  1  10000  0  19  13
20  1  0  0  20  14
21  1  0  0  21  15
22  1  0  0  22  16
23  1  0  0  23  17
24  1  0  0  24  18
BU SAVED IN DISK

ASSY OF PARTITIONED SSM K-AA

MATRIX K-AA STORED IN DISK
ASSY OF PARTITIONED SSM K-ABT
```

```
MATRIX K-ABT STORED IN DISK
CALCULATION OF LA,LB WHEN BV<=48 USING RAM

LA1 STORED IN DISK
DECOMPOSITION OF K INTO L IS COMPLETE
CALCULATE LB USING RAM

CALCULATION OF L INVERTED IS COMPLETE
RUN ASTRA3 TO COMPLETE THE JOB

]LOADS ASTRA3
]RUN

READING INPUT INFORMATION
DOFIU READ IN DISK
BU READ FROM DISK

CALCULATE KC USING RAM
TEST BV= 6
STORE KC1 IN DISK


K-A INVERSE HAS BEEN CALCULATED

CALCULATE NODAL DEFLECTIONS
CALCULATE REACTIONS
DEFLECTIONS HAVE BEEN STORED
PRINT NODAL LOAD, DEFLECTIONS AND REACTIONS
```

NODE #	DOF	NODE LOAD	DEFLECTIONS	REACTIONS
1	1	0	****	-4993.81773
1	2	0	****	-4992.59791
1	3	0	****	-4986.27833
1	4	0	****	195.154713
1	5	0	****	-248.380568
1	6	0	****	176.333181
2	1	0	****	-12.9486432
2	2	0	****	.231835623
2	3	0	****	.465738176
2	4	0	****	27.7020922
2	5	0	****	367.322795
2	6	0	****	559.986885
3	1	0	****	-4993.23362
3	2	0	****	4992.36608
3	3	0	****	4985.81259
3	4	0	****	-195.03738
3	5	0	****	-239.649604
3	6	0	****	165.840146
4	1	10000	1.7884384E-03	0
4	2	0	4.96813565E-08	0
4	3	0	2.68683797E-08	0
4	4	0	-1.68373674E-07	0
4	5	0	3.14832586E-06	0
4	6	0	-2.50380362E-05	0

```
STORE NODAL DEFLECTIONS
RUN ASTRA4 TO COMPLETE JOB

]LOAD ASTRA4
]RUN
PROGRAM ASTRA4


READING INPUT INFORMATION
READ NODAL DEFLECTIONS
CALCULATION OF ELEMENT FORCES

THE MEMBER FORCES WILL BE PRINTED OUT IN FREE FORMAT IN THE ORDER SHOWN BELOW.
FORMAT LIMITATION WITHIN THE BASIC PROGRAMMING LANGUAGE

ELEMT #  NODE #     PX           PY           PZ           MX           MY           MZ

1  1  -8644.48891  5.65790312  -.862542649  71.0760484  165.704804  313.626806
   4   8644.48891  -5.65790312  .862542649  -71.0760484  -76.0667442  274.359735

2  2  -.165394081  1.73377339  -12.8415225  -136.234083  650.021027  90.507976
   4   .165394081  -1.73377339  12.8415225  136.234083  439.61829  56.6075752

3  3  8643.74893  -5.60311923  -1.23664234  69.9909736  186.466418  -288.627799
   4  -8643.74893  5.60311923  1.23664234  -69.9909736  -57.9507782  -293.665432

END OF JOB
```